

# DRAFT SF 298

1. Report Date (dd-mm-yy) 30 April 1996		2. Report Type		3. Dates covered (from... to )	
4. Title & subtitle Department of Defense Technical Architecture Framework for Information Management. Volume 8: DoD Human Computer Interface Style Guide. Version 3.0				5a. Contract or Grant #	
				5b. Program Element #	
				5c. Project #	
6. Author(s)				5d. Task #	
				5e. Work Unit #	
7. Performing Organization Name & Address				8. Performing Organization Report #	
9. Sponsoring/Monitoring Agency Name & Address Defense Information Systems Agency Center for Standards 10701 Parkridge Blvd Reston, VA 20191				10. Monitor Acronym	
				11. Monitor Report #	
12. Distribution/Availability Statement    Approved for Public Release: Distribution is Unlimited					
13. Supplementary Notes					
14. Abstract					
19970212 108					
15. Subject Terms					
16. Report Unclass			17. Abstract Unclass	18. This Page Unclass	19. Limitation of Abstract
					20. # of Pages
					21. Responsible Person (Name and Telephone #)  Marilyn McLaughlin (703) 735-3563

---

Defense Information Systems Agency  
Center for Standards

---

**DEPARTMENT OF DEFENSE  
TECHNICAL ARCHITECTURE FRAMEWORK  
FOR  
INFORMATION MANAGEMENT**

**Volume 8:  
DoD Human Computer Interface Style Guide**



19970212 108

Version 3.0

30 April 1996

DTIC QUALITY INSPECTED 3



## FOREWORD: ABOUT THIS DOCUMENT

This edition of the Technical Architecture Framework for Information Management (TAFIM) replaces Version 2.0, dated 30 June 1994. Version 3.0 comprises eight volumes, as listed on the following configuration management page.

### TAFIM HARMONIZATION AND ALIGNMENT

This TAFIM version is the result of a review and comment coordination period that began with the release of the 30 September 1995 Version 3.0 Draft. During this coordination period, a number of extremely significant activities were initiated by DoD. As a result, the version of the TAFIM that was valid at the beginning of the coordination period is now "out of step" with the direction and preliminary outcomes of these DoD activities. Work on a complete TAFIM update is underway to reflect the policy, guidance, and recommendations coming from these activities as they near completion. Each TAFIM volume will be released as it is updated. Specifically, the next TAFIM release will fully reflect decisions stemming from the following:

- The DoD 5000 Series of acquisition policy and procedure documents
- The Joint Technical Architecture (JTA), currently a preliminary draft document under review.
- The C4ISR Integrated Task Force (ITF) recommendations on Operational, Systems, and Technical architectures.

### SUMMARY OF MAJOR CHANGES AND EXPECTED UPDATES

This document, Volume 8 of the TAFIM, incorporates the following changes from the previous version:

- Chapters 2 and 4 provide more guidance on how to design HCIs, which is the first step in reorientation of the *Style Guide* toward a more process-oriented document.
- Chapters 5 and 6 have had additional material added and the figures updated.
- Minor editorial changes have been made to other chapters.

Future actions with regard to this volume include continuing its evolution toward a "how to design" document, updating the design guidance where needed, exploring methods for compliance, and assessing the impact of the release of Windows95 on the contents of the *Style Guide*. In addition, this volume will be adapted as necessary to reflect the impact of the policy documents and decisions listed above.

## A NOTE ON VERSION NUMBERING

The *DoD HCI Style Guide* went through a number of revisions prior to its inclusion in the TAFIM, and as a result has followed a distinctive version numbering scheme outside of the TAFIM system. Version 2.0 of the TAFIM included Version 3.0 of the *Style Guide*. Version 3.0 of the TAFIM includes a version of the *Style Guide* that would have been Version 3.1, but this volume has been designated Version 3.0 of Volume 8 for harmony with the rest of the TAFIM.

A version numbering scheme approved by the Architecture Methodology Working Group (AMWG) will control the version numbers applied to all future editions of TAFIM volumes. Version numbers will be applied and incremented as follows:

- This edition of the TAFIM is the official Version 3.0.
- From this point forward, single volumes will be updated and republished as needed. The second digit in the version number will be incremented each time (e.g., Volume 7 Version 3.1). The new version number will be applied only to the volume(s) that are updated at that time. There is no limit to the number of times the second digit can be changed to account for new editions of particular volumes.
- On an infrequent basis (e.g., every two years or more), the entire TAFIM set will be republished at once. Only when all volumes are released simultaneously will the first digit in the version number be changed. The next complete version will be designated Version 4.0.
- TAFIM volumes bearing a two-digit version number (e.g., Version 3.0, 3.1, etc.) without the DRAFT designation are final, official versions of the TAFIM. Only the TAFIM program manager can change the two-digit version number on a volume.
- A third digit can be added to the version number as needed to control working drafts, proposed volumes, internal review drafts, and other unofficial releases. The sponsoring organization can append and change this digit as desired.

Certain TAFIM volumes developed for purposes outside the TAFIM may appear under a different title and with a different version number from those specified in the configuration management page. These editions are not official releases of TAFIM volumes.

## DISTRIBUTION

Version 3.0 is available for download from the Defense Information Systems Agency (DISA) Information Technology Standards Information (ITSI) bulletin board system (BBS). Users are welcome to add the TAFIM files to individual organizations' BBSs or file servers to facilitate wider availability.

This final release of Version 3.0 will be made available on the World Wide Web (WWW) shortly after hard-copy publication. DISA is also investigating other electronic distribution approaches to facilitate access to the TAFIM and to enhance its usability.

**This page intentionally left blank.**

## TAFIM Document Configuration Management Page

The latest authorized versions of the TAFIM volumes are as follows:

Volume 1: Overview	3.0	30 April 1996
Volume 2: Technical Reference Model	3.0	30 April 1996
Volume 3: Architecture Concepts & Design Guidance	3.0	30 April 1996
Volume 4: DoD SBA Planning Guide	3.0	30 April 1996
Volume 5: Program Manager's Guide for Open Systems	3.0	30 April 1996
Volume 6: DoD Goal Security Architecture	3.0	30 April 1996
Volume 7: Adopted Information Technology Standards	3.0	30 April 1996
Volume 8: HCI Style Guide	3.0	30 April 1996

Other working drafts may have been released by volume sponsors for internal coordination purposes. It is not necessary for the general reader to obtain and incorporate these unofficial, working drafts.

*Note: Only those versions listed above as authorized versions represent official editions of the TAFIM.*

**This page intentionally left blank.**

# CONTENTS

<b>1.0</b>	<b>INTRODUCTION</b>	<b>1-1</b>
1.1	BACKGROUND	1-1
1.2	PURPOSE	1-2
1.3	COMPLIANCE	1-2
1.4	HUMAN-COMPUTER INTERFACE (HCI)	1-2
1.5	SCOPE	1-3
1.6	INTENDED AUDIENCE	1-5
1.7	DESIGN GOALS	1-5
1.8	ASSUMPTIONS	1-6
1.9	<i>STYLE GUIDE</i> ORGANIZATION	1-7
1.10	BASELINE	1-7
<b>2.0</b>	<b>INTERFACE STYLE</b>	<b>2-1</b>
2.1	STYLE GUIDES	2-2
2.1.1	Commercial Style Guides	2-3
2.1.2	The <i>DoD HCI Style Guide</i>	2-6
2.1.3	Domain-Level Style Guides	2-6
2.1.4	System-Level Style Guides	2-6
2.2	SYSTEM-LEVEL USER INTERFACE DESIGN DECISIONS	2-6
2.2.1	Selecting a User Interface Style	2-6
2.2.2	Deciding on a System-Level Style Guide	2-7
2.2.3	HCI Design Process	2-7
2.2.4	Migration Strategy	2-7
2.2.5	Portability Across Hardware Platforms	2-9
2.2.6	Integration of HCI Environments	2-9
2.3	USING THE <i>STYLE GUIDE</i> TO SOLVE USER INTERFACE DESIGN PROBLEMS	2-9
2.3.1	Selecting a User Interface Style	2-10
2.3.2	Redesigning The HCI to Improve Usability	2-10
2.3.3	HCI Considerations in Selecting Commercial Software	2-11
2.3.4	HCI Considerations in Developing Custom Software	2-12
2.3.5	HCI Design in Tactical Environments	2-12
2.3.6	Migration Considerations	2-13
2.3.7	Portability Considerations	2-13
2.4	UNIFORM APPLICATION PROGRAM INTERFACE (UAPI)	2-14
2.4.1	Introduction	2-14
2.4.2	Range of Approaches to HCI Portability	2-15
2.4.3	Environments Supported	2-20
2.4.4	Considerations for Use of UAPI Tools	2-21
2.4.5	<i>Style Guide</i> Implications	2-27

<b>3.0</b>	<b>HARDWARE</b>	3-1
3.1	INPUT DEVICES AND PROCEDURES	3-1
3.1.1	Pointing Devices	3-2
3.1.2	The Keyboard	3-3
3.1.3	Window Input Focus	3-3
3.2	SPECIAL DISPLAYS	3-3
3.2.1	Flat Panel Technology	3-4
3.2.2	Liquid Crystal Displays	3-7
3.2.3	Gas Plasma Displays	3-8
3.2.4	Electroluminescence (EL) Displays	3-9
3.2.5	Glare-Reducing Techniques	3-10
3.2.6	Large-Screen Displays	3-11
3.2.7	Stereoscopic/3D Displays	3-15
3.2.8	Touch Interactive Devices	3-17
3.3	ALTERNATE INPUT/OUTPUT (I/O) DEVICES	3-18
3.3.1	Visual I/O	3-19
3.3.2	Hearing I/O	3-20
3.3.3	Mobility I/O	3-20
<b>4.0</b>	<b>SCREEN DESIGN</b>	4-1
4.1	INITIAL SCREEN DESIGN FOR ACCESS-CONTROLLED WORKSTATIONS	4-2
4.1.1	Workstation Log-On	4-3
4.1.2	Application Log-On	4-3
4.1.3	Application Log-Off	4-3
4.1.4	Workstation Log-Off	4-5
4.1.5	Initial Workstation Screen Display	4-5
4.1.6	Classification Color Selection	4-6
4.2	SCREEN DESIGN GUIDELINES	4-7
4.2.1	Visual Design	4-7
4.2.2	Functional Screen Design	4-11
4.2.3	Screen Design Standards, Guidelines, and Requirements	4-14
4.3	COLOR	4-19
4.3.1	General	4-22
4.3.2	Color Selection	4-26
4.3.3	Tonal Color Coding	4-30
4.3.4	Color-Coded Symbols	4-30
4.3.5	Map Graphics and Color	4-31
<b>5.0</b>	<b>WINDOWS</b>	5-1
5.1	WINDOW BASICS	5-4
5.1.1	Basic Window Appearance	5-4
5.1.2	Dragging the Window	5-7
5.1.3	Scroll Bars	5-7
5.1.4	Application Area	5-7



5.1.5	Message Area	5-7
5.1.6	Resizing the Window	5-7
5.1.7	Window Controls	5-8
5.1.8	Window Colors/Patterns/Audio Signals	5-8
5.2	WINDOW DESIGN	5-10
5.2.1	General Guidance	5-10
5.2.2	Window Control	5-12
5.2.3	Designation	5-20
5.2.4	Labeling	5-20
5.2.5	Open Window Navigation	5-21
6.0	MENU DESIGN	6-1
6.1	GENERAL	6-2
6.1.1	Consider Response Time and Display Rate	6-2
6.1.2	Instructions and Error Messages	6-2
6.1.3	Explicit Option Display	6-2
6.1.4	Stacking Menu Selections	6-2
6.1.5	Menus Distinct from Other Displayed Information	6-2
6.1.6	Menu Bars	6-2
6.1.7	Pull-Down Menus	6-3
6.1.8	Pop-Up Menus	6-4
6.2	FORMAT	6-5
6.2.1	General	6-5
6.3	HIERARCHICAL MENUS	6-7
6.3.1	Usage	6-7
6.3.2	General Guidance	6-7
6.3.3	Navigating Hierarchical Menus	6-8
6.3.4	Hierarchical Menu Tree Depth and Breadth	6-10
6.4	ITEM SELECTION	6-12
6.4.1	General	6-12
6.4.2	Selection by Pointing	6-14
6.5	MENU OPTION LABELING	6-15
6.5.1	General	6-15
6.5.2	Selector	6-15
6.6	DIALOG BOXES/POP-UP WINDOWS	6-17
6.6.1	Message Wording Guidelines	6-18
6.6.2	Work-In-Progress Window	6-19
6.6.3	Information Box	6-19
6.6.4	Caution/Warning Box	6-19
6.6.5	Menu Box	6-20

<b>7.0</b>	<b>DIRECT MANIPULATION</b>	<b>7-1</b>
7.1	GENERAL	7-2
7.1.1	Hardware Considerations	7-2
7.1.2	Screen Arrangement by the User	7-3
7.1.3	Function Control	7-3
7.1.4	Interaction	7-3
7.2	METAPHORS	7-3
7.2.1	Metaphor Selection	7-4
7.2.2	Metaphor Design	7-4
7.3	ICONS	7-5
7.3.1	Types of Icons	7-6
7.3.2	Icon Usage	7-6
7.3.3	Icon Design	7-9
7.3.4	Design Methodology	7-18
7.3.5	Icon Evaluation	7-19
<b>8.0</b>	<b>COMMON FEATURES</b>	<b>8-1</b>
8.1	TACTICAL SYSTEM COMMON FEATURES	8-1
8.1.1	Date/Time Display	8-1
8.1.2	Latitude/Longitude Display	8-2
8.1.3	User-Definable Parameters	8-2
8.1.4	Wild-Card Characters	8-3
8.2	ON-LINE HELP	8-3
8.2.1	Types of HELP	8-4
8.2.2	General Design	8-5
8.2.3	Accessibility of HELP	8-6
8.2.4	Provide HELP on HELP	8-6
8.2.5	Application Information	8-7
8.2.6	Provide HELP in Context	8-8
8.2.7	User Control of the HELP System	8-9
8.2.8	Provide Consistent HELP Format	8-9
8.2.9	Self-Explanatory and Concise Displays	8-10
8.2.10	Make Return to Application Easy	8-11
8.2.11	Keep HELP Current	8-11
8.2.12	Provide User Options	8-11
8.2.13	System-Initiated Messages	8-12
8.3	INTERACTIVE CONTROL	8-13
8.3.1	General	8-14
8.3.2	Context Definition	8-18
8.3.3	Transaction Selection	8-19
8.3.4	Interrupts	8-22
8.3.5	Error Management	8-24
8.3.6	Alarms	8-26

8.4	FUNCTION KEYS	8-27
8.4.1	General	8-27
8.4.2	Consistency	8-30
8.4.3	Double Keying	8-31
8.4.4	Labeling	8-31
8.4.5	Layout	8-31
8.4.6	Single Keying	8-32
9.0	TEXT	9-1
9.1	TEXTUAL WINDOWS	9-1
9.1.1	Data Field Labeling	9-1
9.1.2	Updatable Fields	9-2
9.1.3	Text Cursor	9-2
9.2	FORM FILLING	9-3
9.2.1	General	9-3
9.2.2	Defaults	9-3
9.2.3	Consistency	9-4
9.2.4	Cursor Movement	9-6
9.2.5	Data Field	9-6
9.2.6	Error Management	9-8
9.2.7	Form Layout	9-9
9.2.8	Labeling	9-12
10.0	GRAPHICS	10-1
10.1	MAPS AND SITUATION DISPLAYS	10-1
10.1.1	General	10-1
10.1.2	Static Display Attributes	10-4
10.1.3	Dynamic Characteristics	10-7
10.1.4	Creating and Editing Map Graphics	10-10
10.1.5	Map Display Characteristics	10-11
10.2	PRESENTATION GRAPHICS (GRAPHS, PICTURES, AND DIAGRAMS)	10-12
10.2.1	General	10-13
10.2.2	Creating and Editing	10-14
10.2.3	Scales, Labels, and Coding	10-16
10.2.4	Identifying Critical Data	10-20
10.2.5	Grid Lines	10-21
10.2.6	Types of Presentation Graphics	10-22
10.2.7	Pictures	10-23
10.2.8	Diagrams (Schematics)	10-26

<b>11.0</b>	<b>DECISION AIDS</b>	11-1
11.1	USE OF DECISION AIDS	11-1
11.1.1	Cognitive Considerations	11-2
11.1.2	External Factors	11-3
11.1.3	When to Use Decision Aids	11-4
11.1.4	When to Consider Alternatives	11-5
11.1.5	Cautions and Limitations	11-5
11.2	DEFINING DECISION AID REQUIREMENTS	11-6
11.2.1	Understand Tasks	11-6
11.2.2	Understand Requirements	11-6
11.2.3	Types of Aids	11-7
11.2.4	Function Allocation Between Humans and Computers	11-7
11.3	FEATURES OF DECISION AIDS	11-8
11.3.1	General Design Considerations	11-8
11.3.2	Provide Decision Alternatives	11-8
11.3.3	Prediction, Simulation, and Modeling	11-8
11.3.4	Identify and Assess Factors Underlying Decisions	11-9
11.3.5	Handling Decision Aid Recommendations	11-9
11.4	USER REQUIREMENTS	11-10
11.4.1	General Considerations	11-10
11.4.2	Decision Aid Interface	11-11
11.4.3	Explanations	11-12
11.4.4	Training	11-12
11.4.5	Decision Graphics and Displays	11-12
11.5	ORGANIZATIONAL FACTORS	11-13
11.5.1	Information Requirements	11-13
11.5.2	Entire Organization	11-13
11.5.3	Complementary	11-13
11.6	FLEXIBILITY	11-13
11.6.1	Change-over Time	11-13
11.6.2	Maintainability	11-14
11.6.3	Type of Support	11-14
<b>12.0</b>	<b>QUERY</b>	12-1
12.1	GENERAL RECOMMENDATIONS	12-3
12.1.1	Ease of Use	12-3
12.1.2	Interactive Queries	12-3
12.1.3	User Assistance	12-3
12.1.4	Error Detection	12-3
12.1.5	Minimum Training	12-4
12.1.6	User-Oriented Designs	12-4
12.1.7	Multiple Search Options	12-4
12.1.8	Appropriate Displays	12-4
12.1.9	Individual Preferences	12-4
12.1.10	Displaying Results	12-4

12.2	QUERY SCREEN DESIGN	12-5
12.2.1	Screen Design Principles	12-5
12.2.2	Query Screen Organization	12-6
12.2.3	Captions (Labels)	12-6
12.2.4	Data Fields	12-7
12.2.5	Data Organization	12-7
12.3	USER REQUIREMENTS	12-8
12.3.1	Search Enhancements	12-8
12.3.2	Automatic Functions	12-8
12.3.3	Word Stemming	12-9
12.3.4	Erasing	12-9
12.3.5	User Satisfaction	12-9
12.4	USER-FRIENDLINESS	12-10
12.4.1	Commands	12-10
12.4.2	Computer Messages	12-10
12.4.3	Error Messages	12-10
12.4.4	Documentation	12-10
12.4.5	Tailor the Interface	12-11
12.4.6	Accelerators	12-11
12.4.7	Backup	12-11
12.4.8	Restore	12-11
12.4.9	Interrupt	12-11
12.5	SEARCHING	12-11
12.5.1	Commands	12-11
12.5.2	Control Functions	12-12
12.5.3	Editing Commands	12-12
12.5.4	Query Formulation Commands	12-13
12.5.5	User-Friendly Searching	12-14
12.5.6	Features	12-14
12.6	MULTIPLE LEVELS	12-15
12.6.1	Accommodate Novice and Experienced Users	12-15
12.6.2	Novice Users	12-16
12.6.3	Experienced User	12-16
13.0	EMBEDDED TRAINING	13-1
13.1	GENERAL	13-3
13.1.1	Initial Use Overview	13-3
13.1.2	Positive User Attitude	13-3
13.1.3	Availability of Embedded Training	13-4
13.1.4	Accuracy of Embedded Training	13-4
13.1.5	Moment of User Need	13-4
13.1.6	Embedded Training Browsing	13-4
13.1.7	Return to the Application from Embedded Training	13-5
13.1.8	Application Restore Screen	13-5
13.1.9	Restore Embedded Training	13-5

13.1.10	Protection from Hazardous or Destructive Actions	13-5
13.1.11	Application Screen Protection	13-5
13.1.12	Noninterference During Critical Operation	13-5
13.1.13	Notification of Critical Operation	13-5
13.1.14	Multiple Stations	13-5
13.1.15	Context Sensitivity	13-5
13.1.16	Consistent Application Interface	13-6
13.1.17	Inconsistent Interface Assistance	13-6
13.2	<b>ADAPTATION TO USERS</b>	13-6
13.2.1	User Control Over Level of Difficulty	13-7
13.2.2	Learning Structure	13-7
13.3	<b>EMBEDDED TRAINING COMPONENTS</b>	13-7
13.3.1	Multiple Components	13-7
13.3.2	Information Database Component	13-7
13.3.3	Reference Component	13-8
13.3.4	Examples Component	13-8
13.3.5	Advisor or Coaching Component	13-8
13.3.6	Common Errors Component	13-8
13.3.7	Record Keeping	13-8
13.4	<b>INSTRUCTIONAL STRUCTURE</b>	13-10
13.4.1	Granularity	13-10
13.4.2	System-Controlled Sequences	13-10
13.4.3	Sequence Control for Experienced Users	13-11
13.5	<b>INSTRUCTIONAL PRESENTATION</b>	13-11
13.5.1	Combined Media Presentation	13-11
13.5.2	Graphics for Method-Based Knowledge	13-11
13.5.3	Reading Requirements	13-12
13.5.4	Advance Organization	13-12
13.5.5	Printing	13-13
13.5.6	Fidelity	13-13
13.5.7	Simplicity	13-13
13.5.8	Verification	13-13
13.6	<b>ACCESSING TRAINING</b>	13-13
13.6.1	Displayed Embedded Training Availability	13-13
13.6.2	Access Via Training Icons	13-13
13.6.3	Structured Menu	13-13
13.7	<b>SCREEN DISPLAY</b>	13-15
13.7.1	Complete Display	13-15
13.7.2	Graphics	13-15
13.7.3	Window Placement	13-15
13.7.4	User Window Control	13-15
13.8	<b>TECHNICAL COMMUNICATION/WRITING STYLE</b>	13-15
13.9	<b>MOBILITY/NAVIGATION</b>	13-15
13.9.1	Mobility Within the Embedded Training	13-15
13.9.2	Embedded Training Navigation Button Display	13-16

13.10	ERROR FEEDBACK	13-16
13.10.1	Immediate Feedback	13-16
13.10.2	Context-Similarity Feedback	13-16
13.10.3	Error Identification	13-17
13.10.4	Tone of Error Message	13-17
13.10.5	System-Initiated Error Feedback	13-17
13.11	ABILITY TO MODIFY	13-18
13.11.1	Additions to the Embedded Training	13-18
13.11.2	Multi-User Systems	13-18
13.11.3	Annotation	13-18
13.11.4	Annotation Search	13-18
13.11.5	Icons Used to Designate Annotations	13-19
14.0	EMERGING TECHNOLOGY	14-1
14.1	PERSONAL LAYER	14-1
14.1.1	Levels of Expertise	14-4
14.1.2	Experienced Users	14-4
14.1.3	Novice Users	14-5
14.1.4	Interaction Styles	14-5
14.1.5	Interface Personalization	14-5
14.2	MULTIMEDIA	14-6
14.2.1	Multimedia Personal Computer (MPC)	14-7
14.2.2	Audio	14-9
14.2.3	Images	14-11
14.2.4	Video	14-12
14.2.5	Text	14-16
14.2.6	Compact Disc Technology	14-16
14.2.7	Authoring Systems	14-17
14.2.8	Design Guidelines	14-20
APPENDIX A	SECURITY PRESENTATION GUIDELINES	A-1
APPENDIX B	GLOSSARY	B-1
APPENDIX C	REFERENCES	C-1

## FIGURES

1-1	NIST User Interface System Reference Model	1-4
1-2	DoD Technical Reference Model	1-4
2-1	Style Guide Hierarchy	2-2
2-2	Different Window "Looks"	2-4
2-3	HCI Design Process	2-8
2-4	NIST OSE Model	2-16
2-5	Example of Native to Native HCI Style Coverage Matrix	2-21
2-6	Building a Portable UAPI Application	2-23
3-1	Large-Screen Display Technologies	3-12
4-1	Sample Workstation Screens	4-4
4-2	Example of How Specific Types of Information Should Be Located on a Window	4-9
4-3	Example of the Proper Location of Display Screen Instructions	4-10
4-4	Tactical Information Display Options	4-12
4-5	Example of Character Size and Spacing	4-17
4-6	Visual Arc Subtended	4-30
5-1	Example of a Typical Windowing Screen	5-1
5-2	Example of the Tiling Approach	5-2
5-3	Example of the Overlapping Approach	5-3
5-4	Window and Screen Related Terms	5-5
5-5	Example of Cluttered Window Design for Tiled Windows	5-13
5-6	Example of Cluttered Window Design for Overlapping Windows Induced by a Complex Background Pattern	5-13
5-7	Example of a Dialog Box Design	5-14
5-8	Example of Different Applications With Separate Menu Bars	5-15
5-9	Example of Figure Animation	5-15
5-10	Example of a Screen Move	5-16
5-11	Example of a Pointer Changing Shape	5-17
5-12	Maximum Size of Window	5-18
5-13	Window Size Too Large, Covering Critical Information	5-18
5-14	Resize Border Removal	5-19
5-15	Example of Active Window Designation	5-21
5-16	Examples of Open Window Maps	5-22
6-1	Example of a Menu Bar	6-3
6-2	Example of a Pull-Down Menu	6-4
6-3	Example of a Pop-Up Menu	6-5
6-4	Example of Logical Ordering of Grouped Options	6-6



6-5	Example of a System-Level Menu	6-8
6-6	Distinction Between Control Options and Command Options	6-9
6-7	Example of a Hierarchical Menu	6-10
6-8	Broad and Shallow Menu Tree vs. Narrow and Deep Menu Tree	6-11
6-9	Example of a Tree Diagram Interface	6-12
6-10	Example of a Combined Mode User Interface	6-13
6-11	Graphic Acknowledgments of Selection Processing	6-14
6-12	Example of Distinctive Subunit Labels	6-16
6-13	Example of Dialog Box	6-18
7-1	Example of Positive and Negative Images	7-2
7-2	Examples of Four Basic Icon Types	7-7
7-3	Example of an Iconic Menu	7-7
7-4	Example of Icons Used on Button Layout	7-9
7-5	Example a Multi-Functional Icon with Interface for Selection of Available Functions	7-10
7-6	Example of Textual Representation	7-10
7-7	Example of a Text Title for an Icon	7-13
7-8	Examples of Icon Shapes	7-14
7-9	Example of the Use of Technological Shapes	7-14
7-10	Example of Mirrored Icons	7-15
8-1	Example of How a Screen Display Can be Customized	8-16
8-2	Example of How a System Can Display Default Status	8-20
8-3	Example of a General List of Control Options	8-22
8-4	Example of a Distinctive Confirm Action, Using a Dialog Box	8-25
8-5	Example of Soft Key Location	8-29
8-6	Suggested Method for Indicating Active and Inactive System Function Keys	8-29
8-7	Recommended Method for a Return to Base-Level Functions	8-30
8-8	Recommended Location for Function Key Labels	8-32
9-1	Example of Interrupt Capability for Multiple Entries	9-4
9-2	Example of How a Paper Entry Form and a Computer Data Entry Form Should Be Consistent	9-5
9-3	Cursor Should Appear in First Character Space of First Data Entry Field	9-7
9-4	Data Entry Should Not Require Leading Zeros	9-7
9-5	Visual Cues for Field Length	9-8
9-6	Example of How Explanatory Messages Can Be Provided	9-10
9-7	Example of a Form Title	9-11
9-8	Example of an Indication of an Optional Field	9-11

10-1	Typical Electronic Map in Black and White	10-2
10-2	Example of Consistent Map Orientation	10-3
10-3	Brigade's Map Display Showing One Echelon Higher and Two Lower	10-4
10-4	Querying a Summary Symbol for Detailed Information	10-6
10-5	Example of a Map Inset	10-8
10-6	Example of How a Computer Model Can Generate Graphics from User Input	10-16
10-7	Texture Patterns	10-17
10-8	Example of Breaks in a Graph's Axes When Scales Have Been Expanded	10-17
10-9	Comparing Distorted Data Trends Induced by Exaggerated Scales to a Proportional Scale	10-19
10-10	Example of Linear Versus Non-Linear Scales	10-19
10-11	Use of Supplementary Text and Actual Values in a Graph	10-21
10-12	Example of an Area Chart	10-22
10-13	Examples of Bar Graphs	10-24
10-14	Example of a Scatterplot	10-25
10-15	Example of a Pie Chart	10-25
10-16	Example of a Graphic Picture	10-26
10-17	Sample Flowchart	10-28
12-1	Examples of Pie Chart, Bar Graph, and Line Graph	12-5
12-2	Data Layout and Justification	12-7
12-3	Sample Text Editing Box	12-13
13-1	Simplified Prototype Embedded Training Screen	13-4
13-2	User Selection of Training Level	13-6
13-3	Example of How an Advisor Can Guide Users	13-9
13-4	Example of How an Advisor Can Guide Users (cont'd)	13-9
13-5	Example of "Cautions" that Identify Common Errors	13-10
13-6	Combined Graphic and Natural Language Presentation	13-11
13-7	Example of Assistance Request Reminder	13-12
13-8	Using an Icon to Access Embedded Training Directly	13-14
13-9	Example of Embedded Training Navigation Buttons	13-16
13-10	Example of Automatic Correction Notification and Identification	13-18
13-11	Example of Identification Annotation Position	13-19

## 1.0 INTRODUCTION

### 1.1 BACKGROUND

The proliferation of computer technology has resulted in the development of an extensive variety of computer-based systems and the implementation on these systems of varying Human-Computer Interface (HCI) styles. To accommodate the continued growth in computer-based systems, minimize HCI diversity, and improve system performance and reliability, the United States (U.S.) Department of Defense (DoD) is continuing to adopt software development standards. The proliferation of new systems and technology in DoD has also made it necessary to continue efforts to develop and provide guidelines for information display and manipulation.

Computer-based system performance and reliability are products of the performance and reliability of individual components. Computer-based system components include hardware, software, and any user involved in the operation, maintenance, or utilization of the system. Of these components, the user is the most important as well as the most difficult to predict. Thus, a key factor of a high performance, high reliability system is an easy-to-use, effective design of the interface between the user, the hardware, and the software.

One contributor to an easy-to-use, effective HCI is standardization. HCI standardization begins with the selection of an accepted Graphical User Interface (GUI), which in turn provides a standard Application Programming Interface (API) and style approach. Traditionally, the GUI has been determined by the software source selected, such as commercial-off-the-shelf (COTS) software, government-off-the-shelf (GOTS) software, or proprietary software applications. The emerging uniform application program interface (UAPI) technology may free the designer from some of this dependence on the software and hardware platform for the interface "look and feel" (see Section 2.0). The variability of users' needs and differing interpretations of GUI style result in the lack of a common approach and the creation of dissimilar HCIs among systems and applications developed by independent organizations. Adding to the problems in standardization is the fact that the commercial GUI styles do not address issues critical to some DoD organizations, such as geospatial systems, map interface controls, acronym standards, security, and symbol shape standardization.

Standardizing the HCI across application software developed within the DoD community is a two-step process. The first step is to define and document the functional goals, objectives, and requirements of the HCI. The second is for the DoD system and application designers to implement HCI standardization.

### 1.2 PURPOSE

The purpose of this *DoD HCI Style Guide* (or the *Style Guide*) is to provide a common framework for HCI design and implementation. Through this framework, the long-term functional goals, objectives, and requirements of the HCI will be defined and documented.

Interface implementation options will be standardized, enabling all DoD applications to appear and operate in a reasonably consistent manner.

Specifying appearance, operation, and behavior of DoD software applications will support the following operational objectives:

- **Higher productivity** - People will accept and use what is easy to understand if it aids them in accomplishing their assigned tasks with minimal confusion or frustration.
- **Less training time** - Standard training can be given once for all applications, rather than requiring users be trained when transferring to new systems or new training be created for each new or changed application.
- **Reduced development time** - It will no longer be necessary to design a complete HCI for each system component, because previously developed *Style Guide*-compliant software will be available. The basic appearance and behavior of the interface will be specified by combining and tailoring the commercial GUI style with guidelines in this *Style Guide*. The specific look and feel of each DoD organization's applications software will be detailed in domain-level style guides. These details will limit HCI diversity and further support for the reduction of HCI development time.

### 1.3 COMPLIANCE

The *DoD HCI Style Guide* has been developed as a guideline document presenting recommendations for good interface design. The *Style Guide* is not intended to be strictly a compliance document; however, it does represent DoD policy concerning HCI design. The interface developer is expected to use the selected commercial GUI style guide, this *Style Guide*, and the appropriate domain-level style guide along with the input of human factors specialists to create the HCI.

The domain-level style guide is the compliance document and may be supplemented by a system-level style guide created as an appendix to the domain-level document. The commercial GUI style guide and this *Style Guide* are expected to be followed in order to maintain consistency and good design principles within DoD. The use of the word "shall" has been eliminated from this document to remove possible conflict of design principles presented with domain-level compliance requirements.

### 1.4 HUMAN-COMPUTER INTERFACE (HCI)

A user is an integral part of a system. The user-machine interface encompasses interactions between the user and the system, including controls, displays, environmental concerns (e.g., lighting, noise), workspace layout, procedures, and documentation. Design of these elements has a major impact on manpower, personnel selection, training, logistics, safety, and human performance, all of which are elements of concern within DoD systems. HCI addresses the user interface as applied to computer-based systems. HCI encompasses the look and feel of the

interface, physical interaction devices, graphical interaction objects, alternate interactions (i.e., voice, touch screen, pen), environmental factors, and any other human-computer interactive methodology. HCI design guidelines in the form of the *Style Guide* provide three major benefits:

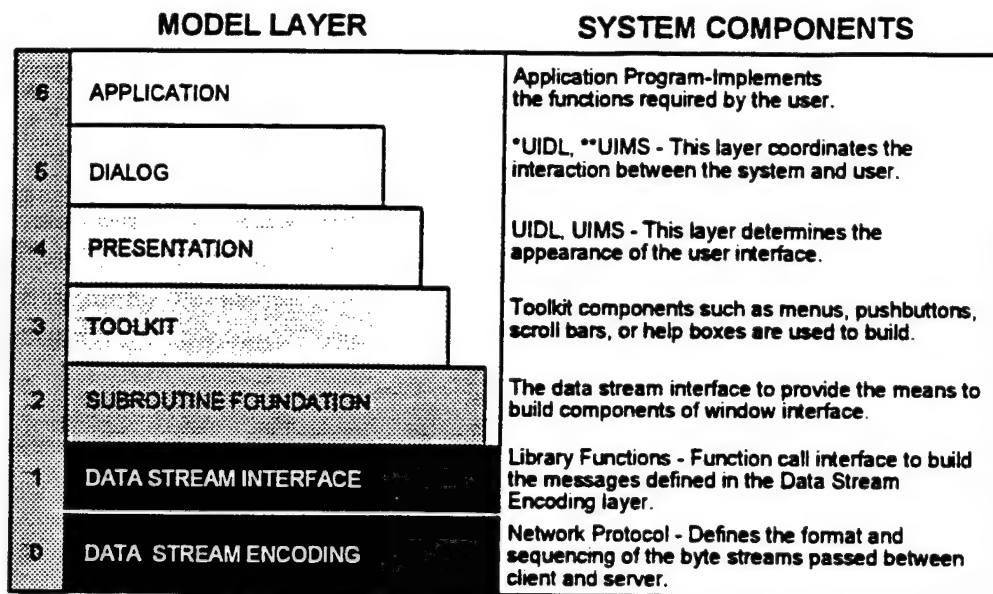
- First, the *Style Guide* along with commercial GUI style guides are resources from which designers may draw to aid in developing usable display screens and interactive procedures. This is especially important because the rapid pace of knowledge acquisition impacts human performance and computer systems, and because the GUI has emerged as the dominant architecture for the HCI.
- Second, the guidelines provide a common approach that supports consistency of design a fundamental principle of human factors engineering design.
- Third, the guidelines will allow for a broader range of personnel selection criteria, and will reduce training and possibly manpower requirements for all systems.

## 1.5 SCOPE

Two factors influence the applicability of the *Style Guide* to DoD computer-based systems: the software architecture being used and the functional requirements of the specific system. This *Style Guide* addresses functional requirements and operations that are intended by DoD to be consistent across the entire interface design. The *Style Guide* emphasis is on HCI considerations for features and functions applicable to DoD systems and applications. Such features and functions include system start-up, security issues, and map graphics.

The *Style Guide* has been developed to address design considerations germane to the DoD environment. The guidelines are generic enough to apply to almost any GUI and, to a lesser extent, to text-based interfaces. The system developer needs to be aware that using a software architecture other than those mandated for use within DoD will limit portability to and reusability by other systems within DoD.

Guidelines are presented for application development within layers 0 through 5 of the National Institute of Standards and Technology (NIST) reference model. Figure 1-1 presents a summary of the NIST reference model. For layers 0 through 2, applications should adhere to the X Window processing standards in Federal Information Processing Standard (FIPS) 158. Layer 3 defines the toolkit standards that support the window management API. Layers 4 and 5 define the look and feel of the GUI. Standards for the upper layers are currently under development by IEEE P1201 committees and may eventually be incorporated into this *Style Guide*. These guidelines define how user interface services are to be provided within the DoD technical reference model (TRM). The TRM, shown in Figure 1-2, defines the set of services to be provided by the application platform and the associated profile of standards for implementing the services.



- \* User Interface Definition Language (UIDL)
- \*\* User Interface Management System (UIMS)

Figure 1-1. NIST User Interface System Reference Model

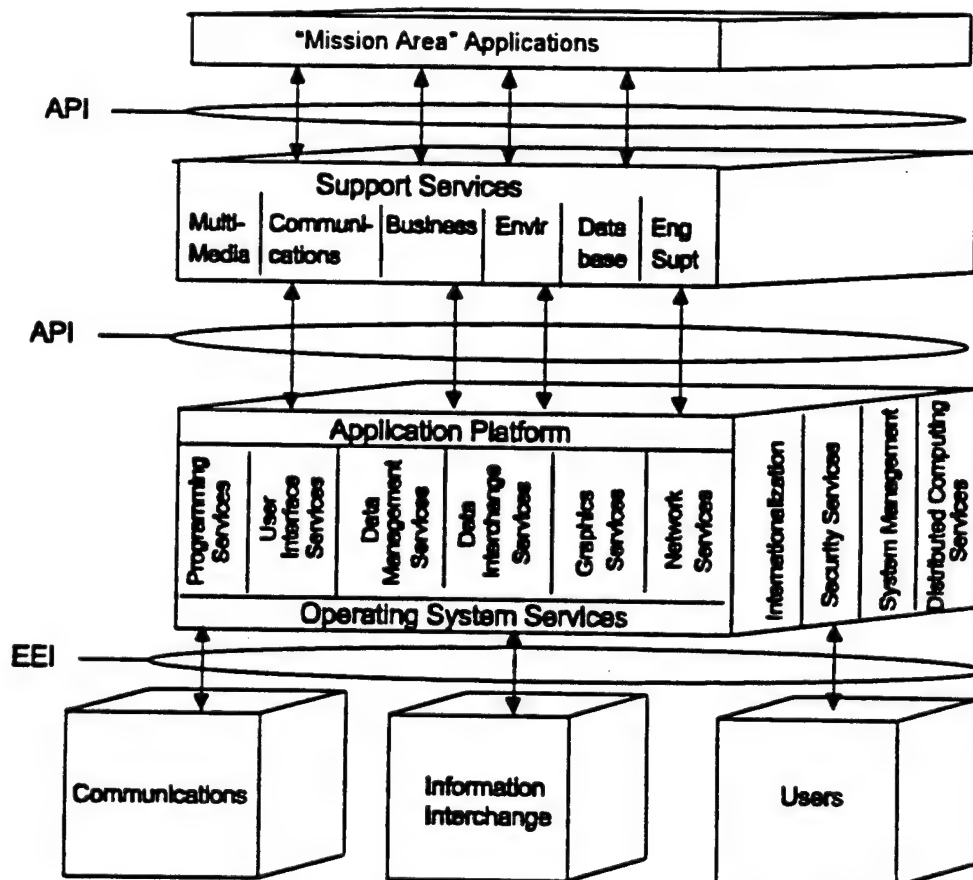


Figure 1-2. DoD Technical Reference Model

## 1.6 INTENDED AUDIENCE

The target audience for this *Style Guide* includes DoD military and civilian personnel along with contractors representing those who determine system requirements, program managers, system managers, software developers, and application HCI designers. Ideally, these individuals should be knowledgeable of the characteristics of the intended user population and the tasks these users must perform. In addition, the users of this *Style Guide* should have some knowledge of human-performance considerations. A secondary audience includes users and software maintainers who are interested in the general design of the interface, who wish to provide feedback concerning modifications and improvements to the *Style Guide*, or who wish to assess the usability of fielded systems or applications in terms of their compliance with *Style Guide* content.

There are two basic environments within DoD that the *Style Guide* addresses: the operational and the business. The operational environment includes both strategic and tactical systems, though not necessarily mission-critical weapons systems. The business environment includes systems used in military and civilian office environments. Systems from within the operational environment are moving towards the use of UNIX, Open Software Foundation (OSF)/Motif, and Open Look for the user interface, whereas the business environment tends to use Microsoft Windows, OS/2 Presentation Manager, and the Apple Macintosh interface styles. A critical difference between the two environments involves the degree of customization that is recommended. In the business environment, with its more stable user community, individual customization is more acceptable. In the operational community, with its higher turnover of users, multiple users, and need for over-the-shoulder viewing, individual customization can have a negative impact on human performance and should be used cautiously.

## 1.7 DESIGN GOALS

DoD application development should:

- First, identify and be familiar with the functions and tasks to be performed by the system and the operational environment. This allows development of an understanding of the overall system dynamics.
- Second, complete an analysis of the capabilities and limitations of system users. A task trade-off analysis between the user and the application is recommended. This allows development of an understanding of which tasks are best performed by the human and which are best performed by the hardware and software. In addition, this understanding provides the groundwork for task and interface design to ensure that the user can successfully perform the required tasks.
- Finally, apply a consistent set of rules for designing the interface. The rules for the design of the HCI include, but are not limited to, the following:
  - Design the applications to meet specific user requirements. Above all, provide the functionality to meet those requirements.



- Ensure that all applications are consistent with the interface guidelines specified in the appropriate commercial GUI style guide, in this *Style Guide*, in the domain-level style guide, and in the system-level specifications.
- Ensure that an application's HCI provides rapid access to all of its functions. To ensure this, avoid unnecessary menus and long selection lists that force users to "page" through all entries.
- Ensure that the application is flexible. For example, provide multiple methods to access a function (e.g., direct command line entry, menus, tree diagrams, mnemonics, and keyboard accelerators).
- Require explicit action to perform any act that could result in irreversible negative consequences, and provide users with options (e.g., quit without saving).
- Give users a choice of input devices (keyboard or pointing device) for scrolling, map manipulation, and invoking or terminating an application. The keyboard and pointing device should be interchangeable where appropriate to the action being performed.
- Ensure that an application user interface does not depend on color to communicate with the user. Color should add substance to the interface, not dominate it.

## 1.8 ASSUMPTIONS

In writing this *Style Guide*, the following assumptions were made:

- The user will interface with information from external systems, COTS software, and GOTS applications.
- The application design requirements specified in this *Style Guide* will be supported by standard DoD civilian computer environments, and tactical or strategic computer environments. The DoD HCI will be implemented on a variety of computer architectures. Computer systems will be equipped with diverse capabilities, such as monochrome versus color monitors and varying amounts of random access memory.
- The *Style Guide* will not address all elements of the human-machine interface. The focus of this document is on the HCI within DoD.
- A system will be composed of a set of applications and will meet the operational needs of users through the integration of multiple applications from a variety of sources (e.g., COTS, GOTS).

## 1.9 STYLE GUIDE ORGANIZATION

Section 2.0 of the *Style Guide* describes the interface style and design issues that must be addressed by software developers within DoD. This section also addresses the concept of application portability between platforms and between GUI styles.



Section 3.0 describes hardware considerations, with focus on input/output devices and their alternatives. This section includes issues related to the Computer/Electronic Accommodation Program (CAP). A subsection on special displays is also included.

Sections 4.0 through 10.0 contain HCI guidelines for the designer. General subjects covered include screen design, windows, menu design, object orientation, common features, text, and graphics. Each section is divided into specific subject areas and includes examples of the stated design guidelines.

Sections 11.0 through 13.0 cover application design guidelines. The topics include decision aids, query, and embedded training. The selected applications represent focus areas of DoD applications and subjects that have generated questions and comments from system developers.

Section 14.0 covers emerging technologies, with initial information on guideline considerations for new areas. This section addresses topics that may become additional sections in later versions or may be added to existing sections.

Appendix A describes objective security interface requirements, using the *DIA Style Guide* and DDS-2600-6215-89 as baselines.

Appendix B, the glossary, defines frequently used terms pertaining to the HCI and GUI style guidelines.

Appendix C, references, is supplemented by direct references at the end of each section, providing a means to determine the original source of a specific guideline. Appendix C demonstrates the overall review undertaken to provide a baseline for this document.

Addenda will describe specific interface requirements of various organizations served by this *Style Guide*. This version of the *Style Guide* includes by reference "User Interface Specifications For The Joint Maritime Command Information System (JMCIS), Version 1.3" as Addendum 1. Additional addenda will be added as required.

## 1.10 BASELINE

The users of this *Style Guide* should seek out the following references for use in interface development:

- *Air Force Intelligence Data Handling System Style Guide* (U.S. Air Force 1990) establishes HCI guidelines for applications developed for Air Force Intelligence analysts and users.
- Blattner, M. M., and R. B. Dannenberg, *Multimedia Interface Design*, ACM Press, 1992.
- The *Defense Intelligence Agency (DIA) Standard User Interface Style Guide for Compartmented Mode Workstations* (DIA 1983, henceforth called the *DIA Style Guide*) and *Compartmented Mode Workstation Labeling: Source Code and User Interface Guidelines, Rev. 1 (Final)* (DIA 1991, henceforth called DDS-2600-6215-91). These documents address

the security portion of the HCI and are intended for designers of applications for compartmented mode workstations (CMW). They outline security-related interface requirements for workstations operating in the system high or compartmented mode.

- The *Department of Defense Intelligence Information Systems (DODIIS) Style Guide*, (DODIIS 1991a) from which Version 1.0 of this *Style Guide* was adapted.
- *DoD Human-Computer Interface Style Guide*, Versions 1.0, 2.0, and 3.0 (1992a, 1992b, 1993), which provide a framework focused on designing the user-computer interface to enhance user performance.
- FIPS 158-1, "User Interface Component of Applications Portability Profile" (NIST 1993), which mandates the use of the X Window protocol, X library, and X toolkit intrinsics.
- Galitz, W. O., *User-Interface Screen Design*, QED Information Sciences, 1993.
- *Human Engineering Design Criteria for Military Systems, Equipment and Facilities*, MIL-STD-1472D (DoD 1989b) and *Human Engineering Guidelines for Management Information Systems*, DOD-HDBK-761A (DoD 1989c), both of which are human factors standards DoD currently uses.
- *Human Factors Guidelines for the Army Tactical Command and Control System Soldier-Machine Interface*, Versions 1.0 and 2.0 (Avery et al. 1990 and 1992), which provide a set of overarching guidelines focused on designing the user-computer interface to enhance user performance.
- "Institute of Electrical and Electronics Engineers (IEEE) Recommended Practices for Graphical User Interface Drivability," Draft 2 (IEEE 1993b, henceforth called IEEE P1201.2). When adopted, this document will standardize those HCI elements and characteristics that must be consistent to facilitate users switching from one look and feel or application to another.
- Kobara, S., *Visual Design with OSF/Motif*, Addison-Wesley Publishing Company, 1991.
- NIST User Interface System Reference Model, as found in *Volume 2: The Technical Reference Model and Standards Profile Summary*, Version 2.0, (Defense Information Systems Agency [DISA], June 1994), has been adopted as the baseline for this document.
- NIST Special Report 500-187, *Application Portability Profile (APP): The U.S. Government's Open Systems Environment (OSE) Profile OSE/1*, Version 1.0, May 1991.
- North Atlantic Treaty Organization (NATO) Standardization Agreement 2019, *Military Symbols for Land Based Systems* (NATO 1990); Army Field Manual 101-5-1, *Operational Terms and Symbols* (U.S. Army 1985b); and "DIA Standard Military Graphics Symbols Manual" (DIA 1990 Draft), which standardize map graphics symbols.

- The *Open Look Graphical User Interface Application Style Guidelines* (Sun Microsystems, Inc., 1990); *Open Software Foundation (OSF)/Motif™ Style Guide*, Revision 1.2 (OSF 1992); *The Windows™ Interface: An Application Design Guide*, Microsoft Press, 1992; *MacIntosh Human Interface Guidelines*, Apple Computer, Inc., 1992, which describe the major X Window GUIs; and MIL-STD-2525, *Common Warfighting Symbolology*, Version 1 (DoD 1994).
- *User Interface Specifications For The Joint Maritime Command Information System (JMCIS)*, Version 1.3 (Fernandes 1993), which defines a common look and feel for Navy command and control systems.

This *Style Guide* draws from the aforementioned documents. The intent is to establish style objectives and guidelines to which all members of the DoD community can transition.

**This page intentionally left blank.**

## 2.0 INTERFACE STYLE

Given the direction of technology development for the HCI, a long-term goal of DoD has been the implementation of a more common, standardized interface style. FIPS 158 (NIST 1990b) was originally implemented to provide guidance to DoD system designer/developers to encourage standardization of the "look and feel" (i.e., interface style) through the use of a common windowing architecture. FIPS 158 was also interpreted to mean that the developer should use either Open Look or Motif as an interface standard to ensure compliance with this standardization. This focus led to the development of earlier versions of the *DoD HCI Style Guide*, which encouraged the use of these styles. Due to changing technology, DoD is now more broadly interpreting the implications of FIPS 158 on interface style. The reasons for this broad interpretation include:

- The emerging capability of other interface styles, such as Apple Macintosh and Microsoft Windows and others, to operate on top of X Window
- The concern for providing guidance for both the operational (e.g., tactical) and business environments within DoD
- The emergence of the UAPI environment tools that allow portability from one computer platform to another.

While X Window provides the underlying technology through which HCI interfaces of different types will achieve standardization and portability, FIPS 158 is now being interpreted to allow for the use of any of the standard interface styles. These interface styles consist of Open Look, Motif, Macintosh, Microsoft Windows, and OS/2 Presentation Manager. This broader interpretation, while providing greater flexibility to interface designers, also increases the potential for reduced consistency/commonality. Therefore, the need for style guides in general, and this *Style Guide* specifically, becomes all the more important. Use of these style guides will ensure that HCIs are developed in accordance with sound principles of interface design and that consistency and commonality are encouraged. The objective of Section 2.0 is to provide the reader with an understanding of both how the *Style Guide* should now be used in interface design and system development, and how the emerging UAPI tools impact system design and the *Style Guide*.

### 2.1 STYLE GUIDES

Good software design requires selecting and using standard practices for various aspects of an application or system design. This helps ensure consistency of appearance and behavior among applications within a system and develops designs to enhance human performance. A number of documents are available that can help the system designer provide guidance to the HCI designer, including standards (e.g., MIL-STD-1472D, DoD 1989b and IEEE P1295, IEEE 1993c), handbooks (e.g., MIL-HDBK-761A, DoD 1989c), and style guides. Of these documents, the style guides may be the most helpful to HCI designers. Several categories of style guides are

available (see Figure 2-1, Style Guide Hierarchy) to a system designer/developer once the specific GUI style has been selected. The style guide hierarchy begins with the commercial style guides and is refined by the *DoD HCI Style Guide*, with specific style decisions given in the domain-level style guide. The detail proceeds from the general "look" of the interface through to specific functionality. Each category of style guide is discussed in Paragraphs 2.1.1 through 2.1.4.

### 2.1.1 Commercial Style Guides

The style guides provided by major software vendors and consortia cover horizontal aspects of effective design, or those aspects applicable to the widest breadth of systems, applications, and domains. Commercial style guides provide standard design practices for specific development environments, such as Motif or Windows. The commercial style guide will provide a broad understanding of how the system will look and, to a certain degree feel, based on the software architecture underpinning the system.

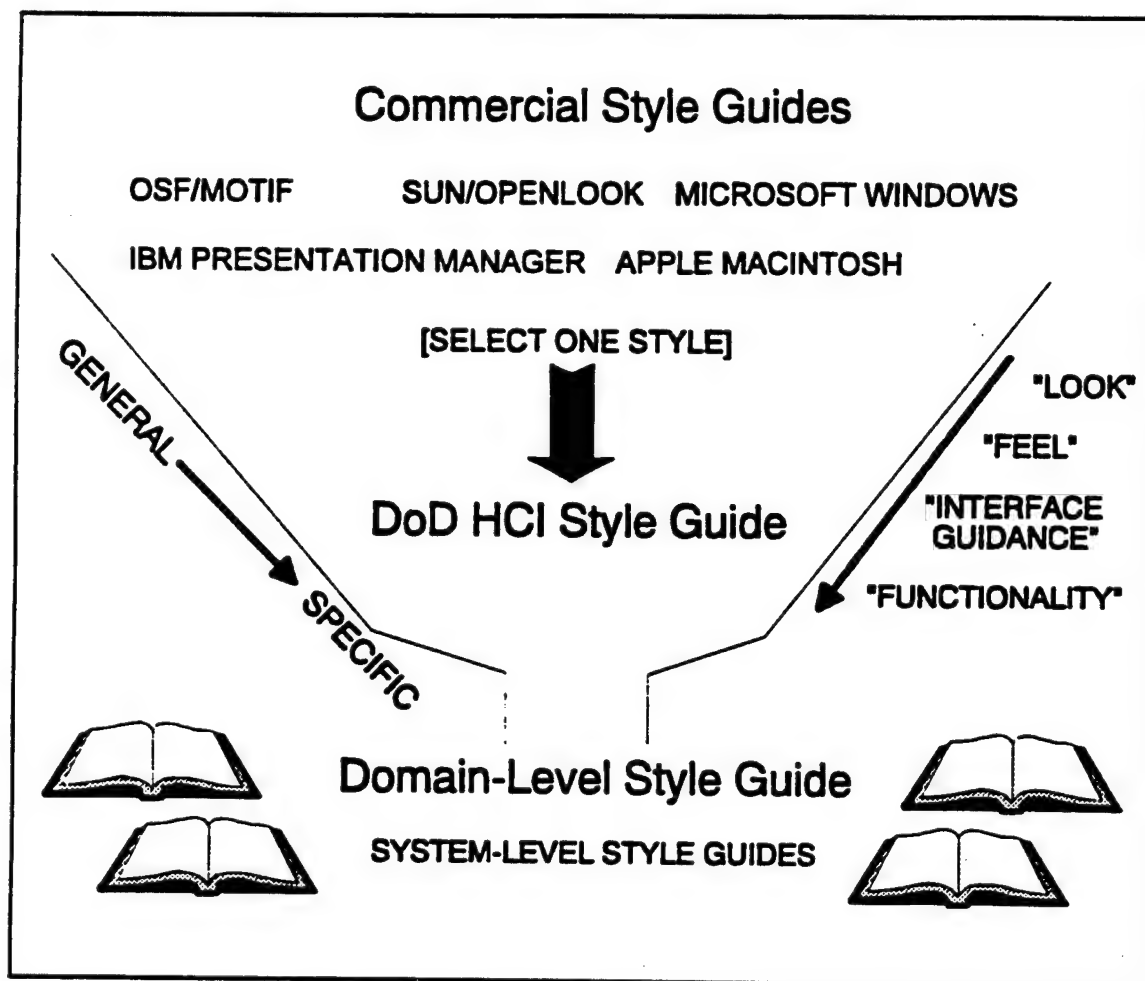


Figure 2-1. Style Guide Hierarchy

Commercial style guides do not necessarily address human performance or military system considerations, but rather more general software behavior. Commercial style guides will provide general guidance that allows a system to deliver a consistent style if a single GUI, such as Motif or Windows, is used. However, the specific style defined by one GUI may differ from that for another GUI, so inconsistencies arise if different GUIs are available on a single platform or workstation.

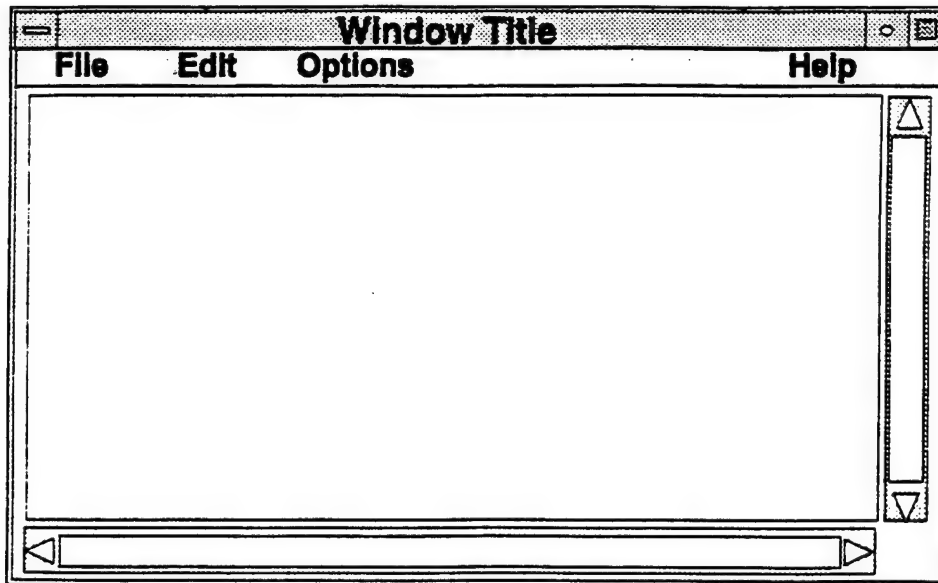
The commercial style guides contain numerous stylistic differences due to different approaches taken by each vendor. These differences can be grouped into the following broad categories:

- **Terminology** - differences in names assigned to, and descriptions of, functions and features. Commercial style guides use substantially different terms to describe the functions and features associated with their respective GUIs. The main distinction is that different terms and descriptive phrases are used to define and describe equivalent or similar functions and features. However, in some instances, the same term is used to refer to different, unrelated functions or features. An example of using different terminology to describe similar functions is: Motif uses the term "radio button" and Windows uses "option button." Both terms refer to lists of selections for which only one choice can be made.
- **Look** - differences in the appearance of displays based upon different styles. The concept of look can be illustrated by comparing the graphic representations (see Figures 2-2a through 2-2d) of each major style.
- **Feel** - differences in the actions a user takes to interact with an application. For example, the differences in the feel of Motif and Windows interfaces are illustrated by the application of keyboard special-purpose keys, mnemonics, and accelerators; and by the use of some special-purpose controls. Both Motif and Windows support keyboard input, but there is very little consistency between the two GUIs in defining special-purpose keys.

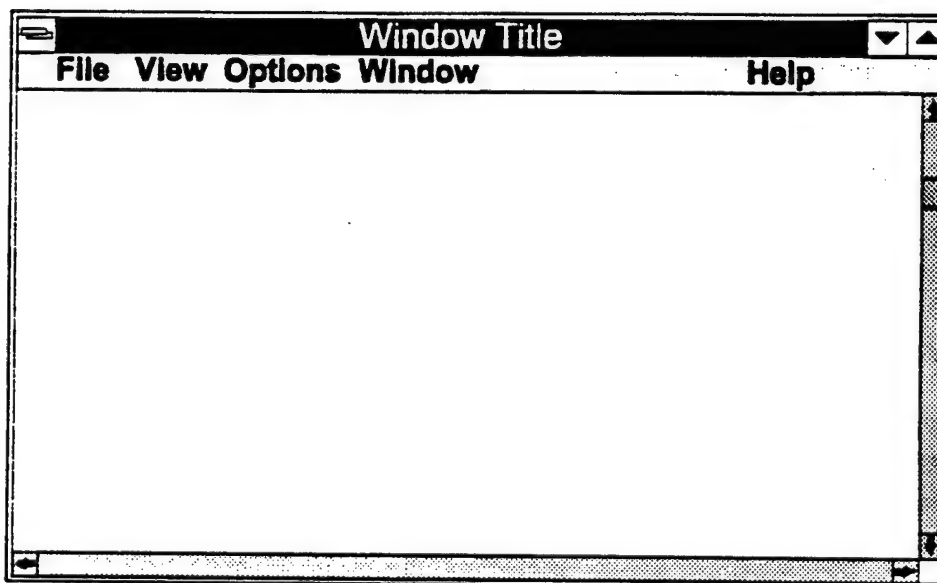
### 2.1.2 The DoD HCI Style Guide

The *DoD HCI Style Guide* provides an additional source of interface design input along with commercial style guides that can be used by a system developer/designer. The *Style Guide* addresses common user interface design issues, contains guidance derived from research on human performance, and provides a focus on elements applicable to DoD systems.

The *Style Guide* promotes consistency by providing generic guidelines that can be applied across the multiple GUIs in use within the DoD environment today. The *Style Guide* provides additional performance-based guidelines for use in designing GUI elements defined within the commercial style guides (e.g., menu and function names, accelerator keys, and mnemonics). The *Style Guide* addresses functional areas applicable to DoD systems not addressed within the commercial style guides (e.g., security classification markings, tactical color codes) and includes appendixes that identify domain-level style guides currently available for the services and other DoD organizations.



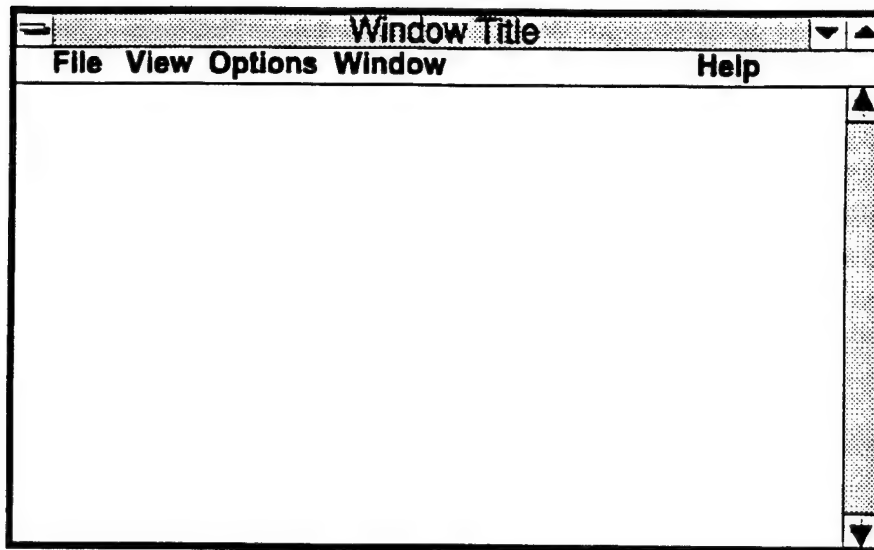
**a. Motif "Look"**



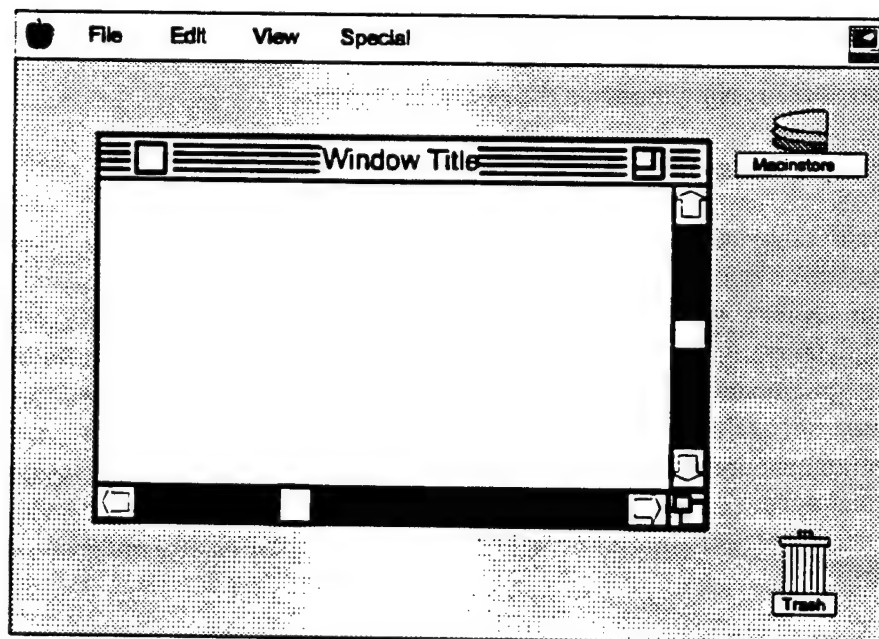
**b. Windows "Look"**

**Figures 2-2 a and b. Different Window "Looks"**





c. OS/2 "Look"



d. Macintosh "Look"

Figures 2-2 c and d. Different Window "Looks"

Beginning with Version 3.1, the *Style Guide* will evolve into a new format focusing on "How To" guidance for combining commercial style guide information with current standards and DoD HCI design considerations. The intent of the new format is to provide complementary information to that available in domain-level style guides, which are intended to provide more detailed "What To Do" specifications. The changes within the *Style Guide* will be staged over the next several revisions due to budget and time constraints. When sections are converted to the How To format, the revised sections will provide specific examples and How To guidance including examples of "good" and "bad" design where appropriate. The successive versions of the *Style Guide* will be reviewed to continue eliminating duplication between it and commercial style guides.

### 2.1.3 Domain-Level Style Guides

Domain-level style guides provide detailed guidance that addresses the requirements of a particular domain (e.g., Command, Control, Communications, Computers, and Intelligence [C4I] and space) as defined by a DoD organization (e.g., joint, individual service, or agency). Domain-level style guides reflect the consensus of the organization on the look and feel they want to provide in their systems. Over time, it is expected that DoD organizations will develop and publish domain-level style guides for directing the HCI design efforts for their systems. An example of a domain-level style guide is the *User Interface Specifications For The Joint Maritime Command Information System (JMCIS), Version 1.3* (Fernandes 1993), which defines a common look and feel for Navy command and control systems.

### 2.1.4 System-Level Style Guides

A system-level style guide, when developed, is used to address system issues and to provide design rules for that specific system. When system-level style guides are used, the look and feel provided in the domain-level style guide is to be maintained. The system-level style guide will provide the "special" tailoring of the commercial, DoD, and domain-level style guides and will include explicit design guidance and rules for the system, as well as document design decisions made during the creation of the user interface. Other style guides may be available from commercial or government sources for a specific application being developed. The system developer should make these documents available to the HCI developer, identify them as reference documents, and call them out in the application-specific technical specification and design documentation.

## 2.2 SYSTEM-LEVEL USER INTERFACE DESIGN DECISIONS

### 2.2.1 Selecting a User Interface Style

The first design decision made for a new system should be the primary style under which the system will be fielded, usually driven by the selection of the hardware and software architecture. The commercial styles most frequently used within DoD include OSF/Motif, Sun/Open Look, Microsoft Windows, IBM Presentation Manager, and Apple Macintosh. However, the preferred style for all DoD tactical applications is OSF/Motif. It should also be noted that the use of Open

Look is discouraged on new DoD systems due to its increasing convergence with Motif and its decreasing use in the general marketplace.

Because the DoD software architecture allows systems to use various commercial styles, the *Style Guide* was developed to address design considerations germane to most style environments. However, regardless of the interface, applications should adhere to the X Window processing standards in FIPS 158.

### 2.2.2 Deciding on a System-Level Style Guide

When required, system-level style guides, with system here defined as a family of applications, represent the tailoring of vendor, DoD, and domain-level guides to meet the special needs of the system being developed. The goal of the system-level style guide is to ensure the development of a standardized, coherent, and usable HCI. A system developer should:

- Select a domain-level style guide, if one is available for the domain and GUI (Assume the domain style guide has evolved from the *Style Guide*).
- Define a system-specific appendix to the domain style guide, if there are system-unique requirements not addressed in that style guide.
- Develop a separate system-level style guide only if an appropriate domain-level document is not available. The system-level style guide should use the relevant commercial style guide and the *Style Guide* as starting points for its content, with tailoring as needed to meet system requirements.

### 2.2.3 HCI Design Process

The system designer/developer should make available all appropriate levels and types of style guides for use in designing the HCI. Figure 2-3 illustrates the process by which a design is evolved from the different types of style guides, in essence moving from the general to the specific. The system concept is then derived from the interpretation of requirements within the guidelines of the standards, style guides, and functionality. While developing the system-level design guidelines and rules, the design should be prototyped as a way to explore and refine concepts with representatives of the user population. This concept exploration will usually help clarify the system requirements and identify aspects of the design or interface style that require special interpretation of the domain-level style guide and/or the creation of a system-level style guide.

### 2.2.4 Migration Strategy

The goal of the DoD migration strategy is to transition existing information-processing systems to a single HCI within an open system architecture. Current DoD policy calls for the HCI to be based on the X Window system in order to provide interoperability among systems. The intent of a DoD migration strategy is to define a generic process that can be applied by all of its systems to achieving this goal.



**Version 3.0**  
**30 April 1996**

### **2.2.5 Portability Across Hardware Platforms**

A critical concern for HCI developers within DoD is how to build an interface on one type of platform and then easily replicate that interface on diverse hardware platforms, either retaining the original interface style or taking on the style native to the new platform while maintaining standardization. A new, emerging technology that may have an impact on this concern and the HCI design process is UAPI. This technology enables the porting of HCI applications from one platform to another and is described in more detail in Subsection 2.4.

### **2.2.6 Integration of HCI Environments**

The integration of business, tactical, finance, personnel, and all other DoD computer interface environments to common HCI principles is a long term goal of DISA. Each of these environments shares common interface issues while at the same time each represents unique interface approaches. The interface guidelines presented in *Style Guide* Sections 3.0 through 14.0 are intended to address common interface issues. The principles of good interface design should be applied to all HCIs used within DoD. The goal of good interface design is to provide the user with the tools needed to complete the required tasks with the greatest ease and effectiveness.

The general difference between the various environments can be described in terms of the usual software within the environment. The business environment is characterized by the use of COTS software as the prime source of application software. The extensive use of COTS software reduces the ability of the system developer to affect the HCI design for the application. The tactical environment has the highest degree of custom-developed software applications, and the result has been the greatest diversity of interface styles and designs. The financial environment carries the legacy of mainframe applications that are oriented to command-line and text-based interfaces. The personnel and logistics applications have the largest databases (other than geographic data) of any of the DoD environments. The maintenance of the database input/output is the focus of these interfaces. The specialized interfaces, such as those used in real time weapon system application, have interface requirements that are beyond the scope of the *Style Guide*. The creation of domain-level style guides is especially important to those systems not completely covered in the *Style Guide*. The general principles given in this document apply to all interfaces, but some specialized areas require separate consideration.

## **2.3 USING THE *STYLE GUIDE* TO SOLVE USER INTERFACE DESIGN PROBLEMS**

Integrating all DoD HCI environments (i.e., business, tactical, finance, personnel) to common HCI principles is a long-term goal of DoD. Each of these environments shares common interface problems, while at the same time each has unique interface requirements. The following paragraphs address the *Style Guide* approach to the common problems and provide guidance for applying the principles so that users are provided with the tools needed to complete the required tasks with the greatest ease and effectiveness.

### **2.3.1 Selecting a User Interface Style**

- a. **PROBLEM:** Many commercial applications in office environments use Microsoft Windows. Additionally, an increasing number of commercial applications are available with either the Motif GUI or with the Apple/MacIntosh GUI.
- b. **RECOMMENDATION:** A single GUI should be selected for use within a work group. Choices include Microsoft Windows, Apple/MacIntosh, OS/2 Presentation Manager, or Motif.

### **2.3.2 Redesigning the HCI to Improve Usability**

- a. **PROBLEM:** The software was not designed to do the task(s) to which it is currently applied. It follows that the labels, headings, and indicators are not consistent with the user requirements.

**RECOMMENDATION:** The interface should be redesigned as soon as possible, because continued use of an inappropriate interface will reduce productivity and lower morale. Sections 6.0 and 9.0 apply to this problem.

- b. **PROBLEM:** The software has been designed to mirror a non-automated (i.e., paper) system without elimination of duplicate inputs, and uses input formats that are not optimized for the computer.

**RECOMMENDATION:** The interface should be redesigned as soon as possible since continued use of an inappropriate interface will reduce productivity and lower morale.

- c. **PROBLEM:** Terminology, jargon, acronyms, capitalization, and abbreviations are not consistent with the users' expectations and common understanding.

**RECOMMENDATION:** These aspects of the interface can cause critical errors in operation and reduce productivity. The software should be revised or upgraded as soon as possible in these circumstances. Sections 8.0 and 9.0 apply to this problem.

- d. **PROBLEM:** The task sequence within the software is not consistent with the operational tasks the operator/user is required to accomplish using the software. In some cases, the use of a software application may take more time and effort than the corresponding manual system.

**RECOMMENDATION:** The requirements/specifications for the software should be reviewed and redesign undertaken, if appropriate. Sections 6.0 and 9.0 apply to this problem.

- e. **PROBLEM:** The application extensively uses data available in other applications, but no interoperability or connectivity is supplied. The operator/user spends large time sequences in duplicate data entry.

**RECOMMENDATION:** The data entry process is error prone and should be minimized where possible. The use of interconnectivity to reduce duplicate data entry is encouraged. Information in Section 9.0 applies to this problem.

- f. **PROBLEM:** The application software employs codes and/or procedures from prior software applications that are difficult to remember but no longer required due to changes in technology.

**RECOMMENDATION:** The interface should be designed to simplify the users' tasks and take advantage of improved technology. The requirement to use cryptic input codes should be eliminated wherever possible.

- g. **PROBLEM:** The software is very complex and requires extensive operator/ user training to make effective use of its capabilities. The result is that the software is rarely or never used, with subsequent loss of the capability offered by the application.

**RECOMMENDATION:** The addition of software navigation aids, improved HELP, and possibly on-line tutorials should be considered in cases where complete redesigns are not cost-effective. Sections 6.0 and 8.0 apply to this problem.

### 2.3.3 HCI Considerations in Selecting Commercial Software

- a. **PROBLEM:** The primary source of application software in a particular domain may be COTS software packages. This may be a problem because the COTS software has a great deal of variability in quality of interface design.

**RECOMMENDATION:** Evaluation copies of proposed software purchases should be subjected to compliance evaluation based upon the domain-level style guide or, if one is not available, the *Style Guide*. This should occur prior to procurement of multiple copies. The procurement of COTS software should provide for the comparison of applications with parallel functionality (i.e., Word Processor with Word Processor; Spreadsheet with Spreadsheet). Comparison should include user evaluation, HCI evaluation, functionality, and compliance with the appropriate domain-level style guide and/or the *Style Guide*.



### 2.3.4 HCI Considerations in Developing Custom Software

- a. **PROBLEM:** The acquisition of custom software introduces nonstandard GUIs into the environment. There is also an increase in the diversity of the HCI look and feel due to stovepipe development if more than one custom system is developed.

**RECOMMENDATION:** The procurement of custom software applications should be required to be in compliance with the applicable domain-level style guide or if one is not available the *Style Guide*. The standard commercial interface style that is used by the domain (environment) that will use the software should be specified for the application unless it is not a GUI. If the interface style normally used is not a GUI, the specification should be directed to an accepted GUI.

### 2.3.5 HCI Design in Tactical Environments

- a. **PROBLEM:** The tactical environment frequently involves operator/users using the same application on the same hardware in shifts. The consistency of look and feel is increased in importance under these conditions.

**RECOMMENDATION:** Compliance with the *Style Guide* and appropriate domain style guide should be combined with compliance to system-level specification and style guide (if needed) to establish as much consistency as possible within and between sets of applications available on the system.

- b. **PROBLEM:** Tactical applications frequently use maps as the basic screen background. Map usage is encouraged but presents difficulties in background foreground contrast, clutter, resolution, and system response time.

**RECOMMENDATION:** These issues must be addressed in HCI design. See Section 10.0 for more information.

- c. **PROBLEM:** The tactical environment frequently has difficulty maintaining the availability of trained operators and circumstantially may require partially trained individuals to operate a given application.

**RECOMMENDATION:** This problem increases the importance of the HELP system, embedded training, ease of operation, and consistency of the interface. See Sections 8.0, 13.0, and 14.0 for more information.

- d. **PROBLEM:** The tactical environment frequently creates a high stress level on the operator/user during use of the application. The high stress environment makes operators more error-prone in their interaction with the application.



**RECOMMENDATION:** Careful attention should be given to error management within tactical applications. See Section 8.0 for information on HELP systems and Subsection 9.2 on form filling.

- e. **PROBLEM:** The use of multiple operators on the same hardware and application requires maintaining a consistent interface.

**RECOMMENDATION:** Although commercial software offers configuration and color choices to the operator, offering the choices is not recommended in cases where multiple operators share the use of the same equipment. See Subsection 4.3 for more information.

- f. **PROBLEM:** The use of color in the tactical environment has preassigned specific meaning.

**RECOMMENDATION:** The use of color and color combination must be carefully planned and controlled in tactical applications. See Subsection 4.3 for more information on color.

#### **2.3.6 Migration Considerations**

- a. **PROBLEM:** The interface is either "command line" or "text based" with the experienced users resisting change and new users requiring extensive training.

**RECOMMENDATION:** The DoD goal is to convert to GUI as soon as possible. However, in these cases, consideration should be given to allowing access to the original interface as a subset of the HCI to provide a transition for experienced users. Sections 5.0 and 7.0 apply to this problem.

- b. **PROBLEM:** The software is different (not consistent) in look and feel from other applications in the same environment.

**RECOMMENDATION:** The goal of consistent look and feel within DoD applications should be a factor in determining application upgrades and replacements. Sections 6.0 and 7.0 apply to this problem.

#### **2.3.7 Portability Considerations**

- a. **PROBLEM:** The software was not designed for the hardware system on which it is being used and contains inappropriate operator actions or is excessively slow in executing commands.

**RECOMMENDATION:** The use of one of the methods for transporting software described in Subsection 2.4 should be reviewed along with an investigation into the cost benefit of upgrading the software and hardware.

- b. **PROBLEM:** Individual users employ more than one workstation or share a workstation with other users.

**RECOMMENDATION:** A personal layer system (see Subsection 14.1) should be created.

## **2.4 UNIFORM APPLICATION PROGRAM INTERFACE (UAPI)**

### **2.4.1 Introduction**

Application program portability from one computer platform to another is an OSE goal for DoD. The advent of GUI technology provided flexibility for HCI design and opened new options, while introducing complications for cross-platform compatibility of application programs. As a result, there is a heightened need for ensuring that consistent GUI software design is planned deliberately and appropriately.

The fact that designers have chosen user interface styles that are compliant with the *Style Guide* to ensure compatibility with X Window has contributed to possible portability of applications. Thus, style restrictions become less of an issue. The FIPS 158 (NIST 1990b) acceptability of alternate development environments broadens as GUI application development environments begin to allow for planned switching from one HCI style to another. This broader view is enhanced as applications are transported from one host platform to another.

These developments have special significance within DoD, given the diverse needs of its two basic environments -- operational and business. The operational environment has a greater need for HCI application interfaces that appear and behave the same, regardless of the host platform. This decreases the need for training and the chances of error by military users within the operational (tactical and strategic) environment, where personnel turnover can be significant. The business environment also has a need for reducing training requirements and human error potential, but this tends to be accomplished more through having the ported HCI applications take on the native platform's look and feel. The user population in the business environment may be accustomed to interacting with the interface style of the particular host platform, and there may be less of a tendency towards personnel turnover. In either type of application, as more HCI applications are developed using UAPI tools, the need continues for designs that support standardization of human performance considerations. The translation of an application should be carefully monitored to avoid the possibility of hybrid GUI styles emerging as an outgrowth of these conversions.

Concepts for UAPIs are currently being developed as industry standards and will represent the basis for commercial GUI development tools that can provide HCI style and application

portability. The IEEE, through IEEE P1201.1 and P1201.2, and the NIST have both addressed the issues of GUI application portability by identifying elements of functional commonality for window system objects and by suggesting the use of development tools to provide insulation from and, at the same time, access to HCI style-specific attributes.

Tools that enable transporting GUI applications across host computer platforms represent relatively new technology -- technology that is rapidly evolving and transforming the software development process. It is the range of options and issues surrounding these GUI development tools and their direction of evolution that are of interest to maintaining HCI style consistency and conforming to human performance style guidance as provided in this document.

The developers of GUI applications need to be aware of the current and evolving state of these tools, as the options available from which to choose are varied by commercial product line, each variation holding the potential for even broader utility in future releases.

#### **2.4.2 Range of Approaches to HCI Portability**

The need to migrate application software across different user interface styles has driven efforts to separate generic window resource functionality from the style-specific attributes and specific window system access requirements. The notion of separation has prompted a review of the conceptual architecture of both applications and application development environments. This has led to multiple approaches to designing a portable GUI application development tool. The approaches are born out of Object Oriented Programming (OOP) concepts, the way that those concepts lend themselves to the GUI problem, and the characteristics of software development.

GUI systems allow the user to interact with a visual representation of an application model. Visually, the components of an application take on the characteristics of "objects," with all the concepts of OOP lending themselves naturally to the graphical interface application description.

OOP design promotes decoupling interface code from code that implements functionality as well as encapsulation of subproblems and their generic solutions into modules. Together, these two concepts provide the basis for changing the architectural view of the GUI application to one that separates HCI style-defining parameters from general window object functionality, allowing generation of new versions (multiple styles) with the same functionality.

The IEEE committee developing the draft standard for "Uniform Application Program Interface (UAPI) -- Graphical User Interfaces" (P1201.1) -- describes an event-driven "model of interaction between the code implementing application program functionality and the code implementing the user interface," with the fundamental UAPI objects, windows, and events combining to form "pre-built, customizable visual objects" or "controls." This system view establishes layers of functionality to allow for flexible modification, expansion and extension, and orderly communication between layers, all of which are prerequisites for portability. However, the layered view of the HCI application includes more than separating the HCI visual object functionality from its look and feel; it includes the fact that the human-computer interface has some depth, and issues of access and integration exist on both sides of the interface.

The DoD Technical Reference Model (see Figure 1-2) has functions at the application platform level that relate to User Interface Services. These services include the following:

- Graphical client-server operations that define the relationships between client and server processes operating within a network, in particular, graphical user interface display processes. In this case, the program that controls each display unit is a server process, while independent user programs are client processes that request display services from the server. See FIPS 158.
- Display objects specifications that define characteristics of display elements such as color, shape, size, movement, graphics content, user preferences, interactions among display elements. See the *Style Guide*.
- Window management specifications that define how windows are created, moved, stored, retrieved, removed, and related to each other. See FIPS 158.
- Dialogue support services translate the data entered for display to that which is actually displayed on the screen (e.g., cursor movements, keyboard data entry, external data entry devices). See IEEE P1201.x.

Figure 2-4 shows the functions required of an application in addition to its functional code, through its HCI services, and how this relates to the NIST OSE reference model. The OSE model includes the application, the API, the application platform (hardware and software), and the interface to the external environment.

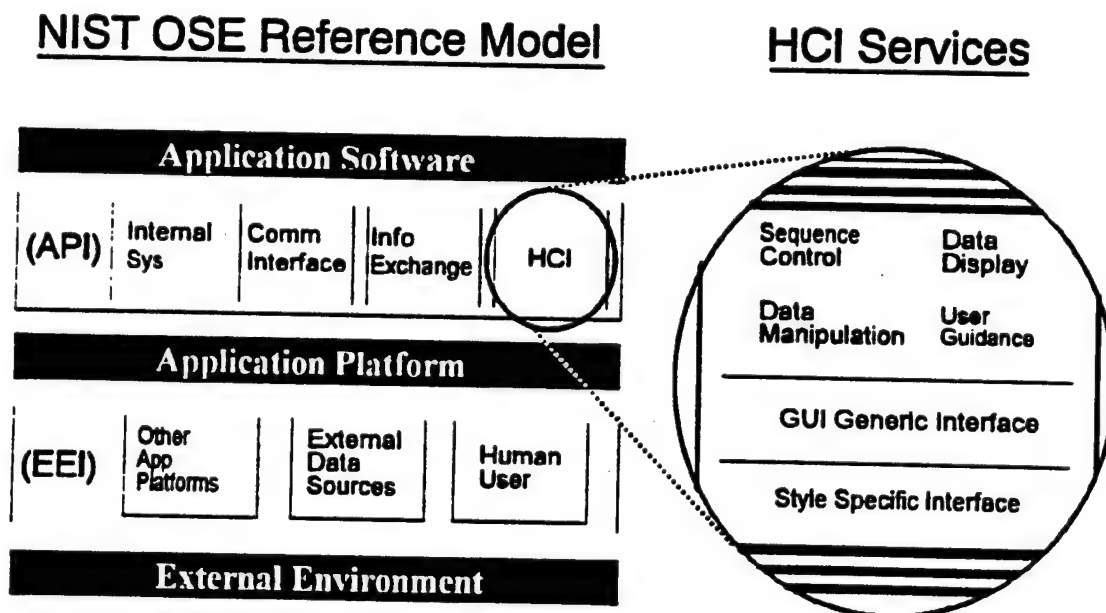


Figure 2-4. NIST OSE Model

The expanded view of the HCI portion of the API depicts an idealized model separating the generic GUI interface object (component) functionality from the specific HCI style's interface and the HCI services of the application code. Identifying and accommodating these HCI services on one side of the interface is equivalent to identifying and accommodating the visual presentation and "direct" manipulation issues on the other side.

The four areas of the human-computer interface services include sequence control, data display, data manipulation, and user guidance, defined in following paragraphs. Since a user interacts directly with these HCI services, they provide part of an outline for the functional requirements of a GUI development tool and road map for evaluating the range of capability such a tool offers.

## DEFINITIONS OF THE FOUR AREAS OF HCI SERVICES:

- **Sequence Control** - Sequence control is defined as the actions taken by the user to direct the computer. Actions involve initiating, interrupting, or terminating a computer process and include the system response to the user's action. System responses to an unsuccessful attempt to control the system are included in the user guidance portion of the user human-computer interface. Common methods of sequence control include command line entry, form fill-in, prompted (question and answer) dialogs, menu selection, function keys, and direct manipulation. Alternative methods of sequence control include voice-entered commands and gesturing. Each method has applicability based on user characteristics (e.g., novice, expert, casual, handicapped), function to be controlled, physical environment, available technology, cost, and other design constraints. Most systems employ a combination of sequence control methods in their interface.
- **Data Entry** - Data entry is defined as the act of entering data into the computer and includes the system's response to data entry. The range of user actions covered by this area of HCI is as varied as there are types of data. Text entry is one of the simplest (reference Section 9.0, TEXT). Other forms of data include graphics, maps, imagery (reference Section 10.0, GRAPHICS), and voice (reference Subsection 3.3, Alternate Input/Output [I/O] Devices) -- each with its own method or methods of entry. Other data types include 3-dimensional data (reference Subsection 3.2, Special Displays), multimedia data, virtual reality, and holographic data. Data entry is accomplished using many of the same methods used for sequence control. For example, direct manipulation (reference Section 7.0, DIRECT MANIPULATION) is often used for graphics data entry; form fill-in (reference Subsection 9.2, Form Filling) is often used for textual data entry.
- **Data Display** - Data display includes not only the display of data entered by the user, but the user's ability to control the data display. Thus, text entry into a word processor is a data-entry task, while changing the visual display attributes from normal to bold text is a data display task. Many data display issues are determining factors in the "look" of a system. These include data density, data location, color, contrast, special attributes, image resolution,

refresh rate, and update frequency. Similar data display issues exist for audio displays. These include volume, tone, pitch, and timbre. The user's method and flexibility to control the data display portion of an application differs according to the type of data being displayed.

- **User Guidance** - User guidance includes feedback to the user for unsuccessful sequence control attempts (e.g., entering an undefined parameter) as well as guidance for unfamiliar features. On-line help, context-sensitive help, on-line tutorials, and error feedback are all examples of user guidance. Error messages are a portion of user guidance but are usually addressed as part of data sequence control.

The need for a GUI development tool to interact with system services to produce a functional application forms another part of the outline for functional requirements and evaluation. The process required to effect transporting that application from one host platform to another and the anomalies encountered upon porting also need to be considered. This process may use the same or a different GUI style (depending on platforms), and the process is conditional on GUI style.

The IEEE P1201.1 view allows implementation methods to vary in scope and approach. This is done by establishing objects and types of human-computer interaction components. The types of interactions required are those that can form a base set without specifying how those interactions will be implemented or what the components look like to the user.

In addition to the draft standard for a GUI UAPI, IEEE draft "Recommended Practice for Graphical User Interface Drivability" (P1201.2) lists characteristics of GUIs that must be consistent to permit users to easily transfer or switch from one look and feel or application to another without causing confusion, requiring retraining, or provoking errors. The defined uses of mouse buttons; the ability to reduce, enlarge, and close windows through title bar icons; and the changes in appearance of disabled/activated choices are examples of drivability issues.

Commercially available portable GUI development tools have handled access and integration between "layers" in a number of ways. The architectural view of the GUI application may be stratified into conceptual layers of functionality, but tools and kits to develop that GUI application have each targeted different parts of the application development problem. These are exemplified in the following two approaches.

#### **2.4.2.1 Toolkits and Class Libraries**

At one level, toolkits and class libraries provide the tailorable GUI components in code form (usually in an interactive graphical form) necessary to build a basic user interface. This speeds development time and helps the developer maintain consistency within the application because it avoids having to construct the individual components from graphics primitives. Extensive programming to link the components into composite/complex window objects and to integrate the HCI interface code into the rest of the application code can be expected with this approach.



And there is no guarantee of portability, unless these components are available in the two-layer form: generic GUI interface object resources and style-specific object attributes.

This toolkit/class library approach can be extended to provide event notifications and query commands to make handling the HCI internal operations (e.g., resizing, moving object location, selection, etc.) conveniently modularized for the developer, saving time and enhancing portability with these generic calls. Modularizing these features and creating generic calls in effect establishes a layered application separating the HCI portion from the application windowing system. To be implemented in the new host window system environment, a library to translate those generic calls to the equivalent host platform calls must be available. Translation libraries must also be available for each type of host platform/HCI style combination supported. This approach is referred to as a "layered API."

Variations on this theme are products that provide a development environment which accesses the native toolkit to create GUI components, and products that supply the equivalent of all native toolkits within the development tool's environment. Some products generate application code templates with the necessary entries to integrate the HCI interface code, and some even assist in integrating data exchange with other software applications/files.

Issues with these products include the following:

- The developer has the flexibility to deviate at will from standard or consistent look and feel within a single application.
- Some toolkit/class library environments do not provide templated code for the API or HCI service links necessary for the application to be functionally complete.

#### **2.4.2.2 Application Framework**

A higher level of approach is the application framework: an integrated object-oriented software development system addressing all the application interface services (HCI, Information Exchange, Communications Interface, Internal System) as well as including development tools needed to produce a portable GUI application. Such a system uses the same layered architectural view, but applies it to the entire application development process, not just the HCI portion of the API. This approach is frequently called a virtual API.

This approach allows a developer to work on a level of abstraction that does not presuppose any "common denominator" of native capabilities during design and development of an application, leaving the emulation of attributes not supported by the host system to the API of the application framework. Application design and code are both insulated from the ultimate target application platform with this architecture, so reuse options and portability are a by-product of design.

More than selecting objects from a set of class libraries, the architectural approach to structuring an application with an application framework involves interacting with a flood of structural, window system, operating system, and network system service class managers as well as GUI development service managers. These managers provide the high-level abstraction of services

available as well as development tools to build application components. This approach is much more comprehensive and, as a result, covers more of the UAPI issues and provides a greater depth of options for addressing HCI issues.

Very few commercial products use this approach, but that does not represent viability of the concepts, only maturity of the technology in today's commercial products. To date, these products provide a development environment for skilled system programmers. While the GUI resources can easily be designed by anyone with minimal programming background, the application design and GUI resource integration must be specified through calls to the abstraction's version of data types and functions, as well as the notifications and query commands for the GUI — *not* a matter for occasional programmers.

This approach is much more comprehensive, and as result, covers more of the UAPI issues and provides potential for addressing HCI issues on a broader "open systems" basis to include across networks, platforms, styles, and languages. Other advantages include a much fuller range of functionality and flexibility in GUI layout and development.

Disadvantages include: large amount of overhead code required, very long learning curve, and high cost. Not all HCI services are fully implemented, and there are not as many platforms supported as with the layered products. These tools do not generate code, so the burden of code organization, GUI code set-up and integration, and event processing code implementation falls on the developer. There is also a real danger of hybrid GUI interfaces developing through poor quality control of conversions, the possibility must be reduced by careful compliance reviews.

Intervening levels of approach target a specific combination of system development subprocesses and products to provide a development tool for each approach. Each level of approach brings more flexibility and greater opportunity for portability. But inherent to this flexibility is the risk of inconsistency and the need to ensure that HCI style guidelines are followed.

### **2.4.3 Environments Supported**

Regardless of the level of approach a specific UAPI development tool uses, it has to contend with issues of portability on the HCI Style level and on the development/host platform operating system level. Mappings describing scope of the portability problem and a means to measure the flexibility of commercial tools can be illustrated by example entries in a coverage matrix, such as that in Figure 2-5.

A family of matrixes can be constructed as in Figure 2-5, which describes application portability from one native HCI style to another native HCI style. Because some tools allow hosting an application with an HCI style different than the one native to either the development platform or the host platform's windowing system (or both), there may be several more matrixes to consider.

The five standard graphic HCI styles (Open Look, Motif, MacIntosh, Windows, and Presentation Manager), the native windowing system HCI style, and the operating system/application platform coverage categories provide the dimensions of the coverage matrix.



Dev\Host	UNIX (Motif)	UNIX (Open Look)	UNIX (Native)	PC (Windows)	PC (PM)	PC (Mac)
UNIX (Motif)	X	X		X		
UNIX (Open Look)	X	X		X		
UNIX (Native)						
PC (Windows)	X	X		---		
PC (PM)					----	
PC\ (Mac)						---

**Figure 2-5. Example of Native to Native HCI Style Coverage Matrix**

For example, an application with a specified HCI style of Motif could be developed on a Macintosh and ported to a Sun workstation or a PC running Windows 3.1. This coverage feature would show up on a matrix listing the variety of development platform/HCI styles versus the same variety of host platforms, but with Motif listed as the HCI style in each case.

By category, there are tools providing UNIX/HCI style to UNIX/other HCI style portability, UNIX to other operating system (OS) portability, and many OS/HCI style to many OS/HCI style portability. Specific application portability requirements will determine the type of tool best suited for a particular development project. However, the issues to consider are that not all tools have the same coverage and that vendor-stated "coverage" may cause the developer to infer capabilities not intended or not available from a particular product. In addition, certain HCI styles have options not directly transferable to other styles. An investigation to establish suitability of a particular tool might be required to ensure conformance with the native style of the GUI as defined in the relevant commercial style guide and tailored as appropriate in the domain-level style guide. The DoD style guide is not the relevant document against which suitability of a tool should be assessed, because it is native-style-neutral.

#### **2.4.4 Considerations for Use of UAPI Tools**

Beyond the issue of coverage, a host of additional considerations, options, and issues should be weighed prior to making a UAPI development tool decision. The use of Ada is mandated by DoD policy. This section frequently refers to "C" and "C++;" however, this does not reflect

upon or change DoD policy concerning Ada. Some of these considerations are listed below and discussed in the following paragraphs.

#### UAPI Tool Selection Considerations:

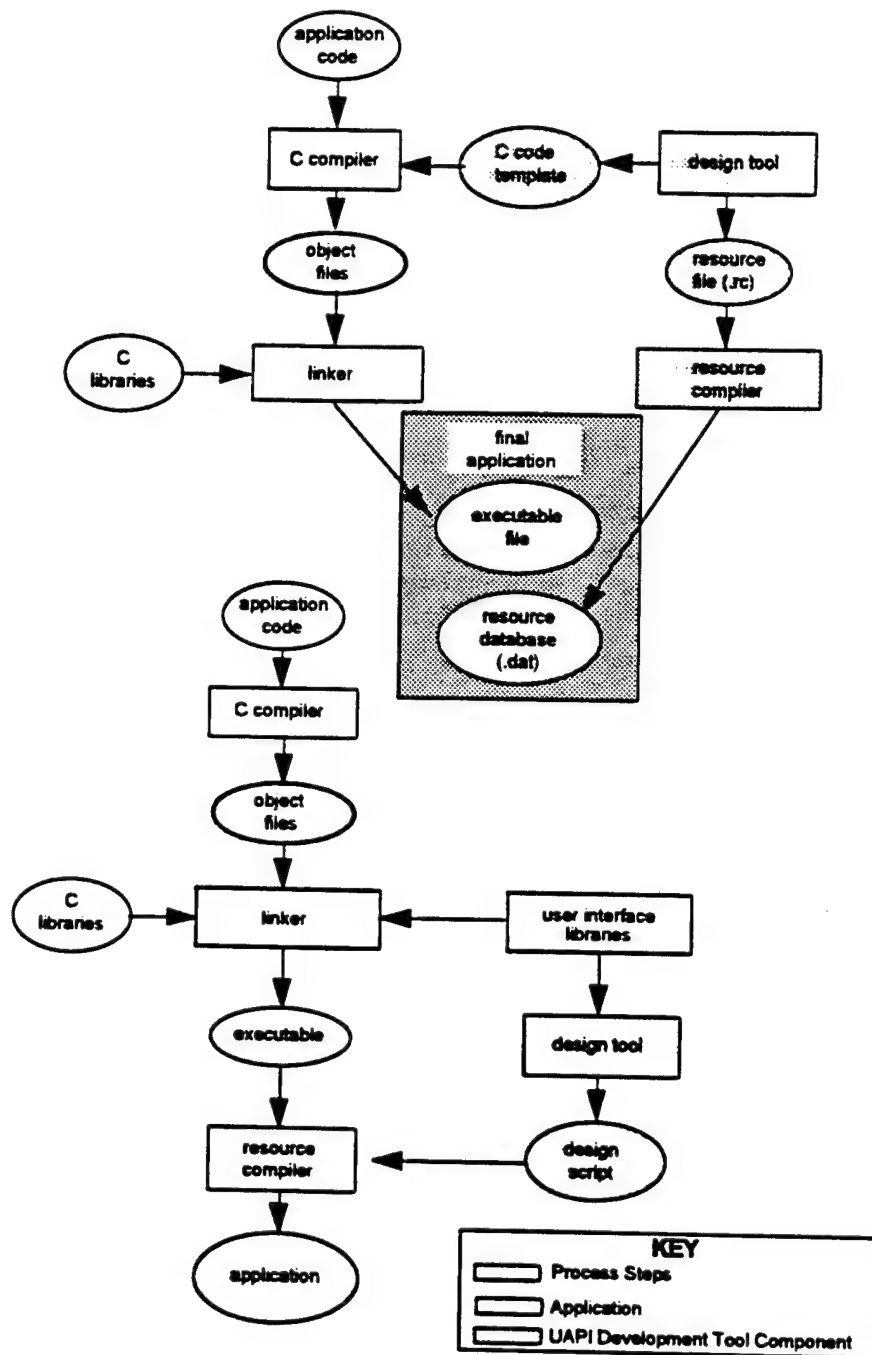
- Portability Implementation/Features
- GUI to Application Code Interfacing
- Interface Requirements to Existing Software
- Window Object Customization
- Tool/Feature Availability/Maturity
- Costs of UAPI Design Tool(s)
- Runtime License Costs
- Designer/Programmer Training Requirements.

##### 2.4.4.1 Portability Implementation

Ideally, an application developed with a portability tool could be introduced to any covered platform and automatically adapt itself to the native OS and windowing system requirements. Once again, ideally, the option of selecting an alternate HCI style could also be made at this time. The state of technology is not quite to that point yet; most of today's tools require the application to be compiled for a specific target platform before the application can be executed in a new host environment. With the exception of application framework systems, the HCI style is not selectable at runtime.

Most layered (toolkit/class library) approaches require platform-specific libraries to be purchased separately and linked or bound to the application, along with development tool-produced resource files for the user interface. Figure 2-6 shows two such layered approaches for building a portable UAPI application capable of being transported to another platform environment. At the top, the native libraries are emulated by the UAPI tool and at the bottom, the native libraries are used directly.

The process involves developing the window object resource files through either an interactive graphical tool or writing the code by hand. The resource file is then associated with the platform-specific libraries (native to the host platform on the lower approach shown in the diagram in Figure 2-6, and tool-supplied on the upper approach) and the functional code. The process is then completed in each case by making the resource compiler-produced binary files available for the application.



**Figure 2-6. Building a Portable UAPI Application**

With both methods, application code must be developed beyond what the UAPI development tool generates. In one method, the development tool produces a C-code template outlining setup and prototyping code for the window object resources, event structure, and other associated files to be completed by the developer outside of the UAPI tool. In the other method, all but the callback specifications are generated, and these may be inserted into the code without leaving the

UAPI tool. Both methods separate the generic functionality of the resources from their HCI style-specific attributes until compiling application/ resource files with style specific libraries. The final application has no real separation of the GUI interface layers.

The choice of target platform and HCI style must be made before or during code preparation and cannot be changed at runtime. However, the end result is the same, as long as there is no need to easily switch back and forth.

Application framework approaches consider a larger context for UAPI application development. Therefore, the designer should allow for HCI styles to be selected at runtime (the separation of layers still apparent), but the target platform must be developer-selected prior to compilation and linking.

Porting the application, once designed and built, can involve specific steps and sometimes additional software products. So the issues here are to find out what the steps and the process are, and ask for specific demonstrations. They vary by product, and product literature does not always provide a clear answer.

#### **2.4.4.2 GUI-to-Application Code Interfacing**

There are no options for consideration here: no commercial tools exist to specify the dialogues between the application and the user interface. Scripting languages and code generators exist in forms suitable for general purposes but fall short of useful as a means to develop the application code that modifies or manipulates data/system states in response to GUI events.

To make the UAPI application functional, the code must be written to integrate the GUI and its functionality. User interface technology has not progressed to the point where suitable design tools are available to support functional application code development for the portable GUIs. Some products are planning to provide a scripting language that appears to assist in this regard. If this represents a trend for commercial products, it will be a significant enhancement to these development tools.

Most UAPI development tools support applications using the development languages C or C++ (with specific vendor products recommended/required), with Ada rarely supported. The process of integrating the functional code and the UAPI requires a C programmer familiar with event-notification programming. With the exception of GUI layout, these tools cannot be effectively used by nonprogrammers.

#### **2.4.4.3 Interface Requirements to Existing Software**

If an application exists, but the requirement to add a GUI to it or port it to other platforms is new, issues become the following:

- Can portable UAPI code be wrapped around existing code to produce a portable product?
- How much effort is required to accomplish the task in either case?

The answers to these questions are subject to semantics and relative to the developer's programming experience and familiarity with the existing code. All of the UAPI development tools reviewed require the functional code to be developed separately as described above, inserting set-up and prototyping code, and tool-unique calls to invoke and manipulate the tool-produced resources at appropriate points. To attach a newly developed GUI, those same UAPI calls would have to be set up, executed, and closed out at the appropriate points in the existing code. Furthermore, most GUI development tools assume a modular software application where each of the routines represent discrete functions of the application. Since this may not necessarily be the case, the existing code may have to be re-engineered in order to insert the GUI code.

The analogy is closer to integration than to wrapping, and the effort involved could be extensive. Given those comments, however, it is possible to take existing code and modify it for portability, assuming that the existing code is in C or C++.

#### **2.4.4.4 Window Customization**

Most UAPI development tools have the option of "subclassing" or creating a new version of an existing window component, which can then be tailored as desired. These tools can also modify the look of a specific object through user overrides of color schemes, icon appearances, and pointer images. Either method allows for window customization and provides the flexibility to extend the window resource set that is supported by the development tool. However, this subclassing for customized objects may limit portability of the application if native capabilities are accessed to design the new features. Some UAPI design tools have utilities to design bitmap graphics and incorporate the customized design into the portable resource set, but not all provide this capability.

Customization may allow violations of style guidance established for the standard HCI styles by creating hybrid interface styles. The use of hybrid styles negates the good human factors and human performance practices outlined in this *Style Guide*. Therefore, this flexibility should be moderated through the domain-level style guide.

#### **2.4.4.5 Tool/Feature Availability/Maturity**

Some products are in the process of refining their capabilities, some are evolving to offer advanced capabilities, and some are aspiring towards well defined future goals. With each release, commercial products gain maturity and realize more of their potential. For specific application requirements, it is best to compare and verify implementation methods, because innovative approaches in implementing new technology may save development time and effort or force an application redesign because of limitations or accommodations.

The basic set of window components as defined by the IEEE UAPI draft document is available in virtually all commercial toolkits. Most have expanded the list of available objects, and offer customization as discussed above and some means of interactive graphic development. Project and file management services are generally available, though not necessarily handy in their current form; and a means to author and integrate custom help files provides an advantage for

some products. Providing the application framework environment to integrate more than the HCI portion of the API will clearly be an advanced step when fully implemented.

Portability to designated platforms can be demonstrated, but product features and services available today are not always clearly separated from target system capabilities and architecture plans in product descriptions. Also, the level of effort in effecting the transportation of an application to a host platform varies by product. At issue is the need to ask very specific questions to determine both the availability of products and the detailed steps required to port applications across host platforms.

It should be noted that there are development tools aimed at a particular application niche -- databases, spreadsheets, knowledge systems -- which contain utilities for creating user interfaces for their particular product. While the GUI interface is not portable in these cases, the total application may be, and it may offer an alternative development option.

Two specific capabilities of interest to operational users within the DoD seem to be conspicuously missing from most commercial UAPI development tools:

- **Cartographic Functions** - Most tactical software applications include requirements for a set of map displays and manipulation operations, such as zoom and jump to coordinates. Special purpose geographic information systems (GIS) provide the map manipulation capability, but do not necessarily interface well with other information systems. Some DoD-sponsored GUI development tools do offer cartographic classes in addition to standard window objects and graphics, but do not offer an interactive graphic development environment.
- **Security Measures** - Operational military users also frequently need to restrict the access of certain subsets of information to authorized users/terminals. OS utilities can provide various levels of security management services across all applications, but are not necessarily portable and do not necessarily offer security services to applications for selective internal use.

Each of these requirements could be built into an application using a commercially available portable UAPI tool and associated application code, but doing so would negate the ease of development offered by using the pre-built resources. Developers should determine if reusable GOTS applications are available that can provide the desired functionality.

#### **2.4.4.6 Costs of UAPI Design Tool(s)/Runtime License Costs**

The range of portable UAPI development tool costs is related to the capabilities bundled together, the architectural approach used for the tool, and the platform on which it will be installed.

For example, a class library/toolkit-based development tool that uses the native libraries might run as much as \$2000 for a particular workstation/HCI Style platform, while its corresponding workstation/HCI style development product might run \$5000. For each platform/HCI style type

(Sun-Motif, Sun-Open Look, HP-Motif, etc.) to which an application is expected to be ported, the specific or separately priced product would need to be purchased. Add to those costs the value of an interactive design tool (\$1500/\$3000 for PC-based/workstation based) if not already integrated.

This type of pricing method offers project flexibility because users pay only for the specific portability options they require. However, for a development tool that substitutes its own complete set of HCI style libraries, the cost is several thousand dollars more per copy (the cost of additional features), and an application framework product may cost up to \$10,000 per copy (the cost of a handling a larger context). Note that some products require the runtime licenses to be purchased separately, although they can be negotiated for purchase in bulk at reduced costs.

#### **2.4.4.7 Designer/Programmer Training Required**

None of the UAPI development tools is intended for nonprogrammers. As noted in this section, most tools target C or C++ development environments and require event-notification programming experience for application development.

Some of the tools handle the set-up and clean-up associated with memory management and window object interactions, and others require the developer to provide the code and insert development library calls appropriately. The sophistication of programmer experience required to complete example applications varies by product, but one characteristic shared by all is that there is a long learning curve before developers are capable of applying and taking full advantage of any of these tools.

Screen layout can be easily accomplished by a nonprogrammer with the GUI design tools available. Some tools require programming to reduce the hazards of inconsistency and nonconformance to the native style as defined by the selected commercial style guide and to the design guidelines in this *Style Guide*.

#### **2.4.5 Style Guide Implications**

The balance between offering the flexibility of many style-specific features on varying platform environments and maintaining consistency in style is a concern for design management. Where development tools restrict some aspects of mixing HCI styles in the same application, they allow others. All of the tools reviewed accommodate custom-developed window objects.

The emergence of UAPI technology will provide a great boon to HCI application developers. While care must be taken to prevent the creation of hybrid GUI interfaces, the benefits for transitions from existing platforms to new platforms or the benefits to training are extensive. UAPIs give the developer a tremendous degree of flexibility in design through:

- Designing an HCI application on one platform and porting it to other types of platforms
- Retaining the pre-existing design or selecting an interface style that is different from the native interface style of the platform to which the porting is targeted



- Assuming the native interface style when porting.

This flexibility also has a burden, ensuring that performance by the user is not compromised through confusing, unique interfaces and increased training requirements. This requires that the flexibility be moderated, to some extent, through standardization of interface styles.

The *Style Guide* and the appropriate domain-specific style guide should be used within DoD to perform this moderation and standardization. These style guides provide the appropriate guidance and framework to guide the developer in tailoring a generic commercial style guides into an application- or system-specific style guide that addresses human rather than software behavior issues, is directed towards DoD design considerations, and presents a more standardized interface style to the user.



## REFERENCES

Paragraph	Reference
2.1.1	Marcus (1992) pp. 187-192
2.4.1	NIST(1991a); NIST (1990b); IEEE (1993a), p. 12, p. 91, p. 93; IEEE (1993b); Valdes (1992a); Hagan (1992)
2.4.2	NIST (1991a); IEEE (1993a) p. 12, p. 90; Murphy (1993); Hagan (1992); Valdes (1992a&b); Goldberg and Robson (1985) pp.1-9; Marcus (1992) p.143-214; Microsoft (1991); Smith and Mosier (1986); Hix (1991); IEEE (1993b) p. viii, p. 1, p.19, p.27-29, p. 35-36, p.49
2.4.2.1	Valdes (1992a); Murphy (1993); Hagan (1992); Meyer (1992); Ga Cote (1992); Karon (1992); Chimera (1993); Hix (1992); XVT (1992); Neuron Data (1992a & b); Visix (1992 & 1993)
2.4.2.2	NIST (1991a); Valdes (1992a); Visix (1992); Karon (1992); Murphy (1993); Chimera (1993); Visix (1993a&b)
2.4.3	IEEE (1993a)
2.4.4.1	XVT (1992); Neuron Data (1992a); Visix (1992)
2.4.4.2	Hagan (1992); Meyer (1992); XVT (1992); Neuron Data (1992a); Visix (1992)
2.4.4.4	XVT (1992); Neuron Data (1992a); Visix (1992)
2.4.4.5	XVT (1992); Neuron Data (1992b); Visix (1993)

**This page intentionally left blank.**

## 3.0 HARDWARE

Hardware refers to the computer and all supporting devices that impact the HCI. It is difficult to develop standard guidelines for all possible hardware variations within DoD, primarily because of the differences in user requirements and the variety of hardware already fielded. Hardware requirements can vary extensively, depending on the function being performed by the system. Some systems are actually information management systems and business systems that do not require immediate user response to information available through the interface. On the other hand, real-time tactical display and control systems require the user to make immediate decisions and input commands from the information on the interface. Each system has different hardware and interface design requirements based on its primary function. The designer needs to understand the selected hardware and the primary function of the system being developed to provide an effective HCI.

Subsection 3.1 will highlight the procedures used to communicate with system applications using a pointing device or the keyboard.

The purpose of Subsection 3.2 is to present guidelines relevant to the specification, selection, use, or design of displays other than the cathode ray tube (CRT). Subsection 3.2 has been further divided into subsections, each of which describes current technology, cites advantages and limitations, and presents available guidelines. Five types of special display technology are:

- Flat-panel displays
- Large-screen displays
- Stereographic and 3D displays
- Glare reduction techniques
- Touch interface devices (TIDs).

Subsection 3.3 is entitled "Alternate Input/Output (I/O) Devices." This subsection addresses the area of nonstandard access to a GUI environment. The guidelines in this subsection consider the requirements of the CAP.

### 3.1 INPUT DEVICES AND PROCEDURES

Subsection 3.1 will highlight the procedures used to communicate with system applications using a pointing device or the keyboard. A comparison is made between Motif and Open Look to illustrate the impact of GUI style selection on hardware. For a more detailed explanation of the input procedures, consult the appropriate GUI style guide.

### 3.1.1 Pointing Devices

A pointing device (e.g., mouse, trackball, tablet, or light pen) allows a user to navigate rapidly around the screen and to specify and select objects for manipulation and action. Throughout this *Style Guide*, the mouse is used as the reference example for all pointing devices.

#### 3.1.1.1 Mouse Button Definitions

Within the DoD community, both two-button and three-button mice are used. For users who must interact with both Motif and Open Look applications, it is important to note that the button definitions and button use differ. The mouse button operations supported by both GUIs are consistent and can be defined as follows:

- **Press** - pushing the mouse button and holding it
- **Release** - letting up on the mouse button
- **Click** - quickly pushing and releasing a mouse button before moving the pointer
- **Double-Click** - pushing and releasing the mouse button twice in quick succession
- **Move** - sliding the pointer without pushing any mouse buttons
- **Drag** - pushing the mouse button and holding it while moving the pointer.

To "drag an object with the mouse" is to move the pointer over the object, press the **SELECT** button on the mouse, move the mouse until the object is in the desired location, then release the **SELECT** button.

#### 3.1.1.2 The Pointer

A key workspace element is the pointer. Objects on-screen can be manipulated by positioning the pointer over an object and appropriately pressing the mouse buttons. The user moves the pointer by moving the mouse on the mouse pad.

**NOTE:** *Different actions are used to move the pointer with other pointing devices, such as trackballs and light pens.*

Pointer shapes provide visual clues to the activity within a window. For example, an hourglass or watch-shaped pointer may indicate that an application is busy, and a cross-hair can be used when sighting on a graphics display.

With the exception of applications using computer-controlled tracking, the pointer should remain where it is placed until moved by the user or the application. The pointer should not "drift."

### 3.1.2 The Keyboard

The keyboard in an operational situation should be virtually interchangeable with the mouse to allow a user to interact with the application by using a pointing device, the keyboard, or both. Business area applications should allow the keyboard to substitute for the mouse but may not find it advisable to provide some keyboard functions through the mouse. Although keyboards vary greatly in number and arrangement of keys, most keyboards include the following:

- **Alphanumeric Keys** - Letters of the alphabet, numbers, punctuation symbols, and text-formatting functions (e.g., Tab, Return, Space Bar)
- **Modifier Keys** - Keys (typically Shift, Control, and Alt) that modify or qualify the effect of other keys (or pointing device inputs) for as long as they are held down
- **Navigation Keys** - Keys that are used to move the cursor (e.g., arrow keys, Home, End, Page Up)
- **Function Keys** - Keys (typically F1 through F10) provided for extra or general functions
- **Special-Purpose Keys** - Keys that have a special function (e.g., Help, Delete, Escape, Backspace, Insert, and Enter).

Because keyboards differ and function keys vary according to application and GUI, a function should not be solely available through a function key. Guidelines for commercial style guide application key assignments are provided in the respective GUI style guides.

### 3.1.3 Window Input Focus

Usually, several application windows are ready to accept input; but only one window, the one with "input focus," actually receives the user input. The window with input focus is known as the active window and is the window where keyboard input appears and pointing device inputs apply.

Most interfaces provide explicit input focus; that is, the user (or application) performs an action (e.g., types appropriate keyboard accelerators, clicks a pointer inside a window, or moves a window to foreground through menu selection) to assign input focus. Implicit focus (the focus is automatically assigned to the window containing the location cursor) is often provided as an option. The default for applications should be explicit focus.

A window with input focus should move to the front of the workspace and be highlighted in some fashion, such as highlighting the window frame or title bar.

## 3.2 SPECIAL DISPLAYS

The CRT is the principal display technology used in computer-based systems. Success of the CRT can be attributed to its ability to inexpensively deliver full-color imagery at high luminance

and resolution. However, tasks of the modern military and emerging alternative display technologies have permitted development of computer-based systems using display technology other than the traditional CRT. Examples include the liquid crystal display (LCD) used in portable computers, and projection technology used to brief military personnel in command centers. As a result, the designer has more alternatives when selecting a display for a military system. The purpose of this discussion is to present guidelines relevant to the specification, selection, use, or design of displays other than the CRT.

Subsection 3.2 is divided further into subsections, each of which describes current technology, cites advantages and limitations, and presents available guidelines. Many reports upon which this subsection is based were published in various conference proceedings rather than in refereed scientific journals and, therefore, may have had less extensive peer review and professional scrutiny. The *Style Guide* user should consult the references for further information on display technologies and human performance considerations.

### **3.2.1 Flat-Panel Technology**

A flat-panel display is flat and light and does not require a lot of power. "Flat" means being thin in form, as well as having a flat display surface. An ideal flat-panel display has the following characteristics: thin form, low volume (cubic size), even surface, high resolution, high contrast, sunlight readable, color, low power, and light weight. Recent advances in flat-panel display technologies have made them realistic alternatives to CRTs for displaying information at computer workstations. Advances have been made in many areas: addressability, contrast, luminance, and color production. Continued research in flat-panel displays has resulted in introducing high information content products that challenge the CRT in specialized applications.

Although there are many different types of flat-panel display technologies, LCDs, electroluminescent displays, and gas plasma displays are the only flat-panel technologies currently mature enough and economical enough to be used in DoD. A major characteristic of each of these display technologies, as distinguished from CRT technology, is that images are formed by turning discrete, non-overlapping, rectangular, cell-based pixels on and off. This discrete, pixel-based structure provides part of the reason that measures of image quality used to evaluate CRT resolution cannot be effectively used to predict image quality and human performance with flat-panel displays.

Factors affecting human performance that differ from the guidance given for CRTs include character-to-character spacing, interline spacing, character and symbol design, the effect of ambient illumination, image polarity, and failure mode. An overriding guideline when specifying flat-panel display technology relative to the CRT is to apply more stringent image quality criteria when selecting flat-panel technology.

### 3.2.1.1 Character Size

Character size is an important variable affecting performance error rates. Height and width of the character and the size of the pixel matrix have important effects on human performance. Exercise special care when determining the character size to use on a flat-panel display.

- a. To improve text search and sorting task performance, use a 9 x 13 pixel matrix or larger.
- b. When displaying dot matrix symbols in nonvertical orientations, use at least an 8 x 11 pixel matrix and preferably a 15 x 21 matrix size.
- c. Character stroke width (SW) should be in the range defined by: (character height ÷ 12) + 0.5 SW (character height ÷ 6). See the following list for guidance.

Pixels in Upper Case Character Height	Minimum Stroke Pixel Count	Maximum Stroke Pixel Count
7 to 8	1	1
9 to 12	1	2
13 to 14	2	2
15 to 20	2	3
21 to 23	2	4

- d. Character height to width should be in the range defined by: (character height x 0.5) character width (character height x 0.9). See the following list for guidance.

Pixels in Upper Case Character Height	Minimum Width Pixel Count	Suggested Minimum Width Pixel Count	Maximum Width Pixel Count
7	4	5	5
8	4	6	7
9	5	6	8
10	5	7	9
11	6	8	10
12	6	9	11
13	6	9	12
14	7	10	13
15 or 16	8	11	14

### 3.2.1.2 Luminance Nonuniformity

Display luminance should be uniform across the surface of the display. Maximum luminance nonuniformity levels should be consistent with the values specified as follows:

Test Object Separation At the Design Viewing Distance	$\angle$ higher $\angle$ lower	Maximum
$>7^\circ$		1,7
5 to $7^\circ$		1,6
4 to $5^\circ$		1,5
2 to $4^\circ$		1,4
$2^\circ$		1,3

### 3.2.1.3 Image Formation Time

Image formation time (IFT) is the time required to render a new image. Four classes of IFTs (see below) have been defined, each relating to information-update requirements for the application. IFTs for all systems should be consistent with Classes III and IV.

Class	Image Formation Time in Milliseconds	Significance
I	$120 < t$	Satisfactory for displays that update an entire page of information at once. Noticeable during key entry. Applications using scrolling, animation, and pointing devices are significantly degraded.
II	$55 < t < 120$	Satisfactory for displays that update an entire page of information at once. Not noticeable during key entry. Applications using scrolling, animation, and pointing devices are somewhat degraded.
III	$10 < t < 55$	Satisfactory for most applications. Motion artifacts can be distracting but are usually acceptable.
IV	$3 < t < 10$	Motion artifacts become less noticeable at formation times approaching 3 milliseconds.



#### **3.2.1.4 Display Failures**

The three most common failures on matrix-addressable displays are cell failures involving individual elements, vertical line failures, and horizontal line failures. Displays can fail actively or passively and leave pixels or lines permanently on or off, respectively.

- Because cell failures often lead to greater performance problems, select displays that minimize the likelihood of cell failure.
- To minimize the performance impact of cell failures, select displays and set display polarity so these failures are likely to match the display background.
- When display element failure is an expected problem, increase the redundancy in the text to minimize the impact on reading performance associated with display element failures.
- Recognition and identification performance with cartographic display is subject to significant decline with as little as 1% pixel failure. Select and maintain displays to ensure a pixel failure incidence below this level.
- Use characters with a pixel matrix larger than 7 x 9 pixels in order to reduce the negative effect of "on" failures.

#### **3.2.1.5 Polarity (Contrast)**

A display with white (or light) characters on a black (or dark) background is said to have "negative contrast" or to be a "positive (image) display." Conversely, dark characters on a light background are said to have "positive contrast" or to be a "negative (image) display." If character stroke width, modulation, and luminance values are nearly equal for both polarities, select a positive contrast/negative image display for better reading speed, search time, and search error-rate performance. The presentation of dark characters on a light background may reduce the effects of reflections on the surface of the display. The effects of glare caused by superimposed reflections are the same for displays of either polarity.

#### **3.2.2 Liquid Crystal Displays**

LCDs are perhaps the most developed and popular flat-panel display technology. Rather than emit light, as do active flat-panel technologies, an LCD controls or modifies the passage of externally generated light. An LCD is typically made of transparent plate electrodes that sandwich a liquid crystal substance. Voltages applied to these electrodes cause realignment of the liquid crystal material, changing its optical properties and allowing light to propagate through the material. By selectively applying voltage to the electrodes, individual display elements can be made light or dark to create the desired image on the LCD.

The LCD is available in a large variety of formats for both commercial and military applications. Display size and resolution range from small, character-based displays (e.g., those in watches) to full-screen computer displays with resolutions to 640 x 480 pixels. LCDs can be monochrome

or color and may operate with backlighting across a wide range of ambient illuminances. LCDs are especially suited for information display in environments where ambient illuminances are high.

Advantages of the LCD include excellent contrast, long life, rugged design, low voltage, and low power consumption (except when backlit). LCD technology is limited by slow speed, limited color capability, temperature range, and manufacturing problems for larger panels with higher resolution.

#### **3.2.2.1 Ambient Illumination**

Provide adequate levels of ambient illumination, because reading performance improves as ambient illumination increases over the range 20-1500 lux.

- Consider LCDs for effective display in high ambient illumination situations.
- In low light situations, provide the ability to adjust the viewing angle and the amount of backlight to enhance the legibility of presented information.

#### **3.2.2.2 Polarity (Contrast)**

For legibility of transmissive or backlit LCDs, use dark characters on a light background (positive contrast/negative image displays). For reflective LCDs, use light characters on a dark background for better performance.

#### **3.2.2.3 Level of Backlighting**

Minimize or eliminate use of backlighting because display reading errors increase as the level of LCD backlighting increases over the range 0-122 candela per square meter ( $\text{cd}/\text{m}^2$ ).

*NOTE:  $\text{cd}/\text{m}^2$  or nit (normalized intensity) is a metric unit for reflected light. One cd or nit is equal to 0.29 footLambert (fL) or one fL is equal to 3.4 nit. A fL is a unit used to measure light reflected from a surface. An ideal surface that reflects all light striking it and diffuses it with perfect uniformity has a luminance of one fL when illuminated by a one footcandle source.*

#### **3.2.2.4 Backlighting and Angle of View**

Carefully consider the potential impact of user performance decrements before using a backlit LCD that is to be viewed off-axis. Backlighting impacts user performance adversely when the display is viewed at an angle.

#### **3.2.3 Gas Plasma Displays**

Plasma panels or gas discharge displays are a widely used flat-panel technology in the information and computer industry because of their inherently high-contrast and high-resolution

capabilities. Images are formed by ionizing a gas, usually neon, trapped between a set of horizontal and vertical electrodes. When an electrical field created by the electrodes is increased rapidly, the gas begins to discharge, resulting in a glow that forms an image. The image can be maintained by sustaining the electrical field, or erased by dropping the voltage below some threshold value. The high contrast exhibited by plasma panels is a result of almost no light output in the "off" state (the electrical field is below threshold) and high luminance in the "on" state. Full-color plasma can be made by depositing phosphors on the glass display surface. The plasma gas discharge in this case excites the phosphor, in much the same manner the electron beam does in a CRT; and color images are produced.

Plasma panels can be found as either monochrome or full-color displays in a number of sizes and configurations. A major advantage of plasma technology is that very bright, high-resolution panels are available. Panels that measure 2048 x 2048 pixels at 100 pixels/inch are available, as well as those that can be viewed in direct sunlight. Panels with luminances of 150-600 cd/m<sup>2</sup> have been produced, with typical large-area, high-resolution display luminances being 30-50 cd/m<sup>2</sup>. Full-color direct current (DC) plasma panels are not yet able to achieve the luminance output nor the display life normally associated with plasma technology.

Features of plasma technology generally include uniformity, high resolution, large size, long life, ruggedness, and absence of flicker. Applying plasma technology in the computer and information industry is limited by high voltage and power requirements, complexity of the drive circuitry, low luminous efficiency, need to develop more fully a color capability, and lack of developers of the technology. Limited information on interface performance is currently available in the literature. In the absence of specific guidance, the designer should use the most conservative approach to interface design.

#### **3.2.3.1 User Concurrence**

Verify the use of gas plasma displays by user personnel as a viable alternative.

#### **3.2.3.2 Testing Prototypes**

Designers and developers of computer-based systems should consider field testing prototypes before committing to gas plasma technology.

#### **3.2.4 Electroluminescence (EL) Displays**

Electroluminescence (EL) displays consist of a layer of polycrystalline phosphor powder or evaporated film phosphors sandwiched between sets of vertical and nearly transparent horizontal electrodes. When an electric field is applied across the polycrystalline phosphor, it is stimulated and light is emitted. The display resolution and the shape of the pixel are defined by the arrangement of the electrodes. EL displays are usually classified as one of four types: either alternating current (AC) or DC thin-film displays, or AC or DC power displays.

AC thin-film displays and DC powder displays are the most advanced of the EL technologies, and discussions of the strengths and weaknesses of EL technologies will be limited to these

types. AC thin-film EL displays have very good luminous efficiency, high contrast, good resolution, and long life. As with some of the plasma technology, these displays require high voltages and complex drive electronics, are expensive, and need research and development to deliver full-color performance. Currently, ELs that display up to 864 lines by 1024 pixels are available for alphanumeric and graphics applications. DC power ELs have a good appearance, simple structure, good luminous efficiency, and the ability to produce gray scales. However, they require high voltages and complex drive circuitry, and have limited luminance output coupled with high reflectance, which may lead to contrast problems. Also, they are expensive. Full-color displays have been produced using DC powder technology, but this area too needs more development. Currently, resolutions compatible with graphics interfaces for personal computers are available (480 lines by 640 pixels) for use in applications that require alphanumerics and moderate graphics. The greatest problems with EL display technology are that developers are few, and investment in research and development is small.

Interface performance information on EL displays is not available in the current literature. In using EL displays, follow the guidelines recommended for LCDs, and use the most conservative level of those recommendations.

#### **3.2.4.1 User Concurrence**

The use of EL displays should be verified by the user personnel as a viable alternative.

#### **3.2.4.2 Demonstration of Concept**

The use of EL displays should be field-prototyped before incorporating into a new system.

#### **3.2.5 Glare-Reducing Techniques**

Glare, as observed on the face of an electronic display, is composed of two components.

1) Diffuse glare or veiling glare, caused by the general illuminance in the environment, can be characterized as a field effect and has little or no modulation. The effect of diffuse glare is to reduce the effective contrast of the display. 2) Modulated or specular glare is the first surface reflection off the faceplate of the display and results from some object or objects in the area surrounding the display. The effect of this type of glare is the appearance of unwanted images on the display surface, making the displayed information more difficult to see and interpret.

The most effective control of glare is to design appropriate workspace illumination so neither diffuse nor specular glare is produced. This is the only method of glare control that will not compromise the resolution and contrast of the display. Because it is not always possible to properly control the sources of illumination, glare reduction techniques have been developed to minimize the unwanted effects of glare.

Many kinds of glare control techniques are used in the electronic display market. Some are screen meshes placed over the display surface, chemical or mechanical etches of the faceplate of the display, anti-reflective coatings, and bonded quarterwave filters. Each has advantages and disadvantages in terms of ability to control diffuse or specular glare, and in terms of effect on

display resolution, flexibility, maintenance, and cost. For example, a bonded quarterwave filter only minimally degrades display resolution but is very expensive, whereas a mesh overlay is very inexpensive but has a major effect on display resolution.

The effectiveness of the glare reduction technique is a function of its ability to suppress each component of glare, while minimizing degradation to the display's resolution and contrast. The desired effect is to match as closely as possible the display performance under optimum conditions. While both contrast and resolution are degraded in an absolute sense, the effective image quality in the operational environment and the acceptance of the display system should improve.

Selecting the glare control alternative most effective for a particular display depends on the information to be displayed, task required of the operator, and environment in which the display will be used. In a command and control facility, use careful analysis and testing to determine the type of glare reduction measures that should be taken.

#### **3.2.5.1 Reflected Glare**

When possible, avoid reflected glare by altering the angular relationship among the observer, display, and glare source. For example, provide the ability to adjust height, viewing angle, and/or contrast.

#### **3.2.5.2 Filter Selection**

When possible, leave selection of the specific glare-reduction technique to individual users.

#### **3.2.5.3 First-Surface Specular Reflections**

Because many types of flat-panel display consist of multiple plates of glass, each of these acts as a specular reflector. All flat-panel displays should incorporate a first-surface treatment to diminish first-surface specular reflections.

#### **3.2.5.4 Etched Filters**

Etches with gloss values of 45 or less should not be used on monochrome CRTs, and etched filters should not be used at all with high-resolution displays.

#### **3.2.5.5 Projection Displays**

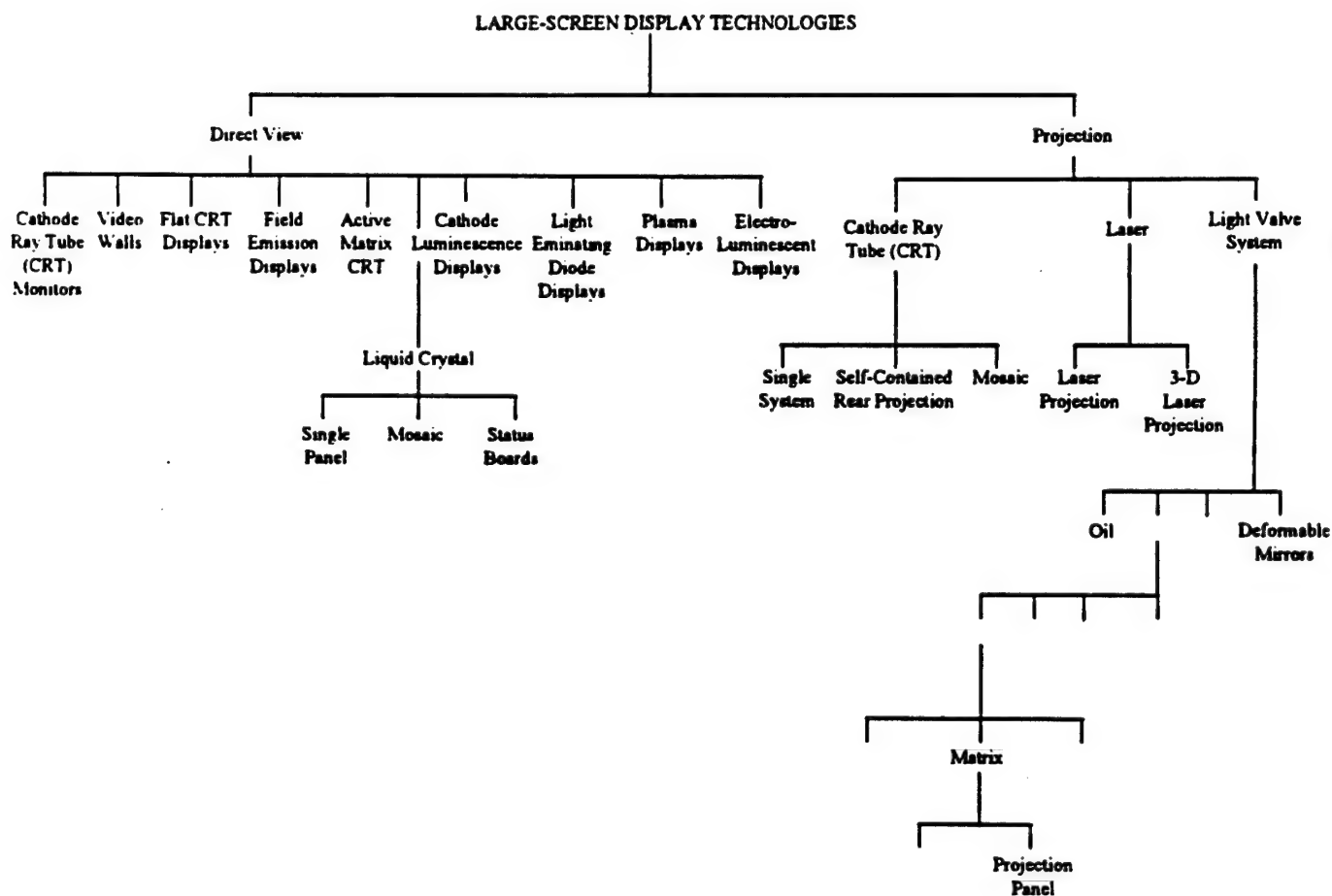
With projection displays, minimize glare potential by positioning projection equipment so the light source is not readily visible to viewers.

#### **3.2.6 Large-Screen Displays**

The DoD operational environment not only imposes requirements for information display at individual workstations, but also for work areas where many persons must observe and use the

information presented on a display. Large-screen displays first appeared in the operational environment as the presentation of mission information on transparent Plexiglas overlays. Today, large-screen technology found in the command center is computer-driven, with the ability to present graphic and video information.

Large-screen display of surveillance, weather, and intelligence information to operations personnel and as a briefing aid to the principal decision-makers is typical in the operational environment. Unfortunately, because of minimal brightness, poorer contrast, and lower resolution when compared to higher resolution desktop displays, most implementations of large-screen technology have been disappointing to the users. Current display technology offers the military a number of choices when implementing a large-screen display. Figure 3-1 illustrates the types of technology available for fielding large-screen displays. The military currently uses both direct view (e.g., high luminance CRTs or large plasma displays) or projection (e.g., light valves or projection CRTs) large-screen displays in its operational facilities.



**Figure 3-1. Large-Screen Display Technologies**

Requirements for large-screen display selection in the operational facility vary relative to use and size of the room in which the display will be used. Large-screen displays found in small briefing rooms (approximately 600 square feet) have a screen size of about 50 square feet, horizontal resolution of 300-1000 pixels, and a luminance output of 300-500 lumens. The briefing room must be nearly dark ( $<2$  foot candles [fc]) when the large-screen display is in use.

The other type of facility where large-screen displays may be found is in the command and control center. The command center may be as large as a two-story structure larger than 2400 square feet, where the information is presented to 10-100 personnel. Large-screen displays in this environment have a screen size of about 100 square feet, screen luminance of 10 fL, horizontal resolution of 800-1000 pixels, and a luminance output in the range of 1000 lumens. Room illuminance in the command center is adjustable from about 5-15 fc, but a more normal office illuminance of 75 fc is often requested. Full-color capability is required of large-screen displays in both facilities. Large-screen displays that are larger, brighter, and have more resolution are desired for the command and control environment.

Selecting or designing a large-screen display, especially a projection display, may be more complex than for other workstations. The effects of ambient illuminance, observer location, and type of data to be displayed are critical in implementing large-screen display technology. Presentation requirements for data not only relate to one's visual acuity for symbol size and contrast when dealing with projection technology, but also to screen size, screen format, symbol luminance, and screen gain. For example, as screen gain increases, the ability to view the screen off the center line decreases. However, a certain amount of screen gain may be necessary to present an image of the necessary contrast, given the expected ambient illuminance in the room. In addition, symbols, graphics, and text should be designed to compensate for the degraded viewing conditions that may exist in the operational environment due to a number of factors. Implementing large-screen displays in the operational environment should always take into account the environmental factors, as well as the information display requirements.

*NOTE: Typical office ambient level is greater than 75 fc, whereas typical command and control centers are 5-15 fc.*

#### **3.2.6.1 Character Dimensions**

Because information is often viewed off the center axis, use character sizes between 10 and 20 minutes of visual arc, with a minimum of a 10 x 14 dot matrix format. When legibility is important, the minimum character height should be 16 minutes of visual arc.

#### **3.2.6.2 Stroke Width**

Ensure that the ratio of character stroke width to character height is 1:6 to 1:10. Use characters with double stroke widths in situations requiring off-axis, longer distance, and/or viewing under difficult lighting conditions.



### **3.2.6.3 Luminance**

Ensure that modulated output luminance, spatially averaged over the full screen, is 300-400 lumens for small conference rooms and command posts and 750-2000 lumens for a command center, assuming 20-40 fc ambient lighting in each case.

### **3.2.6.4 Size versus Luminance**

To ensure legibility, small characters (5 min. arc) require contrast ratios of 15-20:1, and large characters (>20 min. arc) require contrast ratios of 1.5-5:1.

### **3.2.6.5 Aspect Ratio**

Aspect is the ratio of horizontal to vertical dimensions of a character or image. Ensure that character aspect ratio is approximately 1.33:1.48 (width:height ratio).

### **3.2.6.6 Modulation Depth**

Ensure that a display delivers at least 15% visual contrast when measured as modulation depth  $[(L_{\max} - L_{\min}) / L_{\max}]$ , when an alternating pixel pattern is displayed at normal luminance levels.

- Ensure that contrast ratio between the reflected luminance of the screen with a projected light source and the reflected luminance of the screen without a projected light source is approximately 500:1.
- Use positive contrast (black characters on a white background).

### **3.2.6.7 Displayed Data Characteristics**

- Avoid displaying too much data. As with standard displays, consider data type, amount, and appropriate sequence of presentation in designing large-screen display screens.
- If displaying color-coded targets, use only a neutral color such as gray for the background.

### **3.2.6.8 Projection Equipment**

- Minimize glare potential by positioning the projection equipment so that it is not readily visible to viewers.
- To minimize optical distortions, ensure that image source equipment and the projection screen are fully parallel. Electronic or optical distortion-compensating devices may be used to compensate for any remaining distortion and to assure clarity of displayed information.
- Consider using rear projection or other direct view large-screen displays when increased contrast demands are encountered and/or when there is a need to position personnel in the field of projection.



### 3.2.7 Stereoscopic/3D Displays

Displaying 3-dimensional (3D) images and graphics is an emerging technology that may benefit future military applications. Examples where 3D technology may be used are in battlefield and theater of operations analysis, photo-interpretation, teleoperation, air space control, and training and simulation exercises. The goal of proposing 3D displays is to improve user performance and increase naturalness of the interaction. Most current 3D technology is experimental and, as such, is not suitable for an operational environment, although a few stereographic and true 3D electronic displays can be purchased commercially.

Because traditional display technology is a 2-dimensional medium, it has not been able to take full advantage of the human visual system to interpret complex spatial data. Binocular depth information, such as vergence or horizontal disparity, normally in the scene, are not available in the traditional electronic display. Compensation for this has been accomplished by using monocular cues, such as interposition, shading, and perspective. However, improvements in naturalness of the display and potential for gains in human performance with computing systems have stimulated development of systems that make use of the stereoscopic capabilities of the human visual system.

Three-dimensional display technology is classified as stereoscopic and autostereoscopic. The major criterion distinguishing stereoscopic displays from autostereoscopic displays is that the latter requires no special viewing aids to see the 3D image. There is also a difference in the amount of information necessary to create the 3D image.

Stereoscopic displays create a 3D image by requiring an observer to wear a pair of glasses that provides separate images to the left and right eyes. When alternate fields are presented to the eye sequentially at the appropriate rate, the illusion of depth is created. The temporal phase difference that accounts for the stereopsis creating the 3D illusion is usually implemented by requiring the viewer to wear a pair of glasses containing either shuttered lenses, polarized lenses, or red and green lenses. By synchronizing the image presentation to the operation of the glasses, images corresponding to the left and right scenes are presented to the viewer and the illusion of depth is created.

Autostereoscopic displays, by contrast, can be viewed directly. These displays generally use a multiplanar approach to add depth to a 2-dimensional image. Examples of this type of 3D display are BBN's SpaceGraph 3D Display System (uses a flexible mirror to provide the z axis), Tectronix liquid crystal shutter 3D display (uses LCD technology together with a CRT display to create a 3D effect), and Texas Instruments' Omniview (uses a rotating multi-planar surface to produce the z axis). Holography has also produced 3D images, but none to date has been created in real-time.

While innovative technology to provide 3D images is becoming available, no clear guidance outlines where stereoscopic displays might best affect task performance or subjective image quality. Additionally, no database derived from applied vision or human factors research currently exists for developing application guidelines for this new technology. Consequently, the system designer must be cautious in applying 3D display technology in the military

environment. Current technology often limits field of view, number of observers, and type of data that can be presented. It also may exacerbate visual deficiencies that normally have little effect on task performance. Guidance presented here is by no means complete. Many questions remain unanswered, both in terms of human visual response to artificially generated depth from electronic displays and the ways best to enhance performance using this technology.

#### **3.2.7.1 Purposeful**

Presenting 3D information must be purposeful to benefit the user. That is, 3D displays should be associated with the type of work to be performed and required for task completion.

#### **3.2.7.2 System Performance**

Presenting 3D or depth information should not slow information updates, degrade other aspects of system performance, or degrade image quality.

#### **3.2.7.3 Interocular Crosstalk**

Interocular crosstalk or bleed-through occurs in stereoscopic displays when images intended for the left eye are seen by the right eye and vice versa. Because this compromises the observer's ability to fuse the image and perceive it as a 3D object, ensure zero interocular crosstalk between the two images.

#### **3.2.7.4 Color Coding**

Avoid saturated primary colors, as these colors may evoke depth perceptions that may be inconsistent with stereopsis, affecting the perception of depth. Designers should use secondary colors rather than saturated primary colors in coding stereoscopic images.

#### **3.2.7.5 Symbols**

When displaying symbols, ensure that disparity ranges from 0 to 20 minutes of arc in both crossed and uncrossed directions.

#### **3.2.7.6 Dynamic Depth Displays**

When using dynamic depth displays, ensure that the temporal modulation of stereopsis is approximately 1 Hertz (Hz) to ensure the most accurate perception of stereo-motion.

#### **3.2.7.7 Depth-Coded Objects**

Spatially separate depth-coded objects in stereoscopic images to eliminate disparity averaging, crowding, or repulsion.

### 3.2.7.8 Size Scaling

Scale image size to improve the perceived disparity of the image. When accurate size perception is critical to task performance, scale image size for an individual observer.

### 3.2.7.9 Luminance

Because brightness is also a depth cue (bright objects are viewed as nearer), co-modulate luminance with stereopsis.

*NOTE: Display parallax is the apparent displacement of an object displayed on a curved CRT screen and viewed through a flat touch interactive device (TID).*

## 3.2.8 Touch Interactive Devices

A TID is an input device that permits a user to interact with a system by pointing to objects on the display. The TID is considered here because some implementations of touch technology can severely degrade quality of the displayed image. Degradation in image quality using TIDs may be a result of decreased display luminance, reduced display resolution due to visibility of conductors or the device material, increased susceptibility to glare, and dirt on display surface as a result of touching the display surface. Display parallax, caused by separation between touch surface and touch targets, may also contribute to problems with implementing TIDs.

There are six basic types of touch-screen display technologies, each having an impact on display parallax, transmissivity of light, and glare. Each is briefly discussed below. The designer needs to be aware of advantages and disadvantages of each type of TID when selecting hardware and designing interfaces using TIDs.

- Fixed-wire TIDs place wires, either in parallel or in grid fashion, in front of the display. Finger contact with the wire(s) signifies the x,y coordinate of the user's response. This technology is associated with minimal parallax, 70-80% transmissivity, and a medium to high degree of TID glare.
- Capacitive TIDs consist of a transparent conductive film on a glass overlay. Touching this surface changes the small electrical signal passing through the surface, and this signal is converted into the corresponding x,y coordinate. This technology is associated with minimal parallax, 85% transmissivity, and a medium degree of TID glare.
- Resistive membrane TIDs are "sandwich" devices in which a touch results in the contact of two conductive layers. Specific current and voltage levels are associated with individual x,y coordinates. This technology is identified with minimal parallax, 50-60% transmissivity, and a high degree of TID glare.
- Infrared (IR) or light-emitting diode TIDs use IR transmitters along two perpendicular sides of the display frame and photocell receptors along the opposite sides of the frame. A user

touch breaks the resulting matrix of light beams, and the appropriate x,y coordinates of the touch are thus determined. This technology is associated with no parallax problems in seeing the display (although a noticeable degree of parallax exists between the plane of the IR grid and the screen surface for touch responses), 100% transmissivity, and no TID-related glare.

- Surface acoustic wave TIDs operate in similar fashion to IR TIDs, except that the matrix overlay is one of ultrasonic sound beams rather than IR beams. Another approach, "reflective array," uses a piezoelectric transmitter and a series of reflectors and receivers. Touch x,y coordinates are determined by differential timings in reception of the acoustic waves. At least some devices require glass overlay screens. This technology is associated with minimal parallax, 92% transmissivity, and a medium degree of TID glare.
- Pressure-sensitive devices use strain gauges mounted between the display screen and an overlay. Output voltages of these strain gauges are encoded into the appropriate x,y coordinates. This technology is associated with minimal parallax and zero TID glare. Figures for transmissivity are not applicable because the overlay is built into the display screen.

#### **3.2.8.1 Parallax**

Minimize TID/display parallax because it has been shown to lead consistently to poorer entry time and touch count performance.

#### **3.2.8.2 Specular Glare**

Minimize specular glare for applications using TIDs.

### **3.3 ALTERNATE INPUT/OUTPUT (I/O) DEVICES**

The focus of HCI design literature and research has been on the software, displays, physical environment, and computer equipment aspects of the interface. Approaches to testing and evaluating HCIs are usually based on the machine rather than on the human portion of the computer interface. Perceptual characteristics of the expected user are rarely investigated, and interface design ignores known population perceptual limitations. Using color to transfer information does not take into account the potential of color-deficient vision problems in user populations. Using auditory codes does not take into account potential hearing deficits by frequency and adjust outputs according to known population characteristics. The distribution of visual acuity within the user population is usually not considered. It is more likely that environmental impacts on the system will be defined than will user perceptual characteristics.

Accessibility of computer-based systems by persons with disabilities is U.S. Government policy based upon Public Law 99-506 and Public Law 100-542. Individuals with limited hearing, vision, or mobility require enhancements to existing computer-based systems in order to use these resources effectively. These laws address the requirement that acquisition and

management of FIPS resources be conducted in a manner that ensures employees with disabilities access to computer and telecommunications products and services. The implementing regulations for these laws are contained in the Federal Information Resource Management Regulation (FIRMR), 41 CFR Chapter 201.

The interface designer must identify computer and telecommunication accessibility requirements for current and prospective users. The functional aspects of user requirements are an important part of system design and implementation. When automated information environments offer the needed flexibility, users with limitations in vision, hearing, or mobility are ensured full access and integration at a level equivalent to users without disabilities. Flexibility can be achieved in most information environments through off-the-shelf "drop in" or "add on" hardware and software enhancements that modify the common input (e.g., keyboard or mouse) or output (e.g., monitor or printer) interactions associated with computer operations. In addition to being more user-responsive, input capability may need to offer portability, speech input, or wireless connection to the computer. The output may need to be enhanced by magnified text or synthesized speech.

### **3.3.1 Visual I/O**

#### **3.3.1.1 Low Vision**

The term "low vision" covers a broad range of possible conditions and types of visual impairment. The following techniques will enhance conditions for the visually impaired user.

- Use glare protection technology to minimize visual fatigue associated with glare on a monitor.
- Use monitors from 19 to 25 inches to allow increased character size and provide a larger image display.
- Use software or hardware to present images on a computer monitor in a large format. Character sizes can be increased.
- Use software to modify the print size on graphic printers.
- Allow the user to select color schemes for aspects of the application that do not involve coding or status. Ensure that a default scheme is easily available to restore the interface for subsequent users. The control of color will allow the user better contrast control.
- Code the keyboard tactilely with raised dot or bleb. Keycap labels with larger letters can be added.
- Allow the user to select font styles. Ensure that a default scheme is easily available to restore the interface for subsequent users.

### **3.3.1.2 Blind**

The user with very limited or no usable vision will require additional interface enhancement.

- Provide an interface for the visually impaired by using speech recognition technology for input and speech synthesizers for output.
- Hardware and software are available to convert standard word-processing documents so they can be printed on a Braille printer. Dynamically displayed Braille is available on output devices. Braille note-takers' devices are available that are capable of Braille input and output to a personal computer (PC).
- The use of an optical character recognition (OCR) device can translate printed material to speech or Braille formats.

### **3.3.2 Hearing I/O**

#### **3.3.2.1 Visual Redundancy**

Ensure that information conveyed by beeps or speech during computer-related tasks is also displayed visually for the user unable to benefit from the auditory information.

#### **3.3.2.2 Amplification**

Ensure that auditory output from the computer interface has adjustable volume and frequency range.

#### **3.3.2.3 Signaling**

Ensure that alerts and other signals related to the interface are presented in another modality, such as tactile or visual.

### **3.3.3 Mobility I/O**

In addition to the computer interface, review the entire work environment for barriers to access. A variety of interface solutions are available for users with various degrees of limited mobility.

#### **3.3.3.1 Keystroke Input**

Modify the interface hardware and/or software to allow for sequential rather than simultaneous keystrokes. Create keyboard macros to reduce the number of keystrokes required. Adjust the repeat rate of keys, and modify to user requirements the pressure required to activate keys.

### **3.3.3.2 Keyboards**

Alternate keyboards are available that may be more easily used by mobility-impaired individuals. Devices that replace keyboards (e.g., muscle switches, optical pointers, sip and puff systems) are also available.

### **3.3.3.3 Speech I/O**

Using speech recognition technology for input and speech synthesizers for output will provide an interface for the mobility-impaired user.

### **3.3.3.4 Pointing Devices**

The interface should not be dependent on pointing devices and should have redundant input/output capability through the keyboard. The selection of pointing device should consider the mobility parameters of the intended user.

### **3.3.3.5 Optical Character Recognition (OCR)**

Using an OCR device can allow the translation of printed material to a speech-compatible interface.

**This page intentionally left blank.**



## REFERENCES

Paragraph	References
3.1	DISA/CIM (1992)
3.2	Benson and Farrell (1988); Lloyd et al. (1991); Dye and Snyder (1991); Decker et al. (1991); Reger et al. (1989); Biberman and Tsou (1991)
3.2.1	Tannas (1985) page 11; Goode (1991); Beaton and Knox (1987)
3.2.1.1	Reger et al. (1989); Petrun et al. (1985)
3.2.1.1a	Laycock (1985)
3.2.1.1b	DoD (1989a) par. 5.2.6.8.3
3.2.1.1c	ISO (1991) Part 3 Addendum
3.2.1.1d	ISO (1991) Part 3 Addendum
3.2.1.2	ISO (1991) Part 3 Addendum
3.2.1.3	ISO (1991) Part 3 Addendum
3.2.1.4	Lloyd et al. (1991)
3.2.1.4a	Dye and Snyder (1991); Lloyd et al. (1991)
3.2.1.4b	Lloyd et al. (1991); Laycock (1985)
3.2.1.4c	Lloyd et al. (1991)
3.2.1.4d	Dye and Snyder (1991); Lloyd et al. (1991)
3.2.1.4e	Lloyd et al. (1991)
3.2.2	Snyder (1980); Goode (1991)
3.2.2.1	Petrun et al. (1985); Payne (1983)
3.2.2.1a	Payne (1983); Muracka et al. (1989)
3.2.2.1b	Biberman and Tsou (1991); Reger et al. (1989); Cristensen et al. (1985)
3.2.2.2	Kubota et al. (1988); Kubota Murushige Takabayashi and Kobayashi (1986)
3.2.2.3	Payne (1983)
3.2.2.4	Payne (1983); Muracka et al. (1989)

## REFERENCES (cont'd)

Paragraph	References
3.2.3	Tannas (1985) Chapter 10; Goode (1991); Snyder (1980)
3.2.4	Snyder (1980); Goode (1991)
3.2.5.1	Reger et al. (1989); Petrun et al.(1985)
3.2.5.2	Morse (1985)
3.2.5.3	ISO (1991) Part 3
3.2.5.4	Hunter (1988)
3.2.5.5	Woodson et al. (1992)
3.2.6	Breen (1987); Breen (1989); Hurley and Hoffberg (1991); Carbone and MacIver (1987)
3.2.6.1	McNeese and Katz (1986); Shurtleff et al. (1982)
3.2.6.2	Woodson et al. (1992)
3.2.6.3	Blaha (1990); Breen (1989)
3.2.6.4	Shurtleff et al. (1982)
3.2.6.5	Woodson et al. (1992)
3.2.6.6	Blaha (1990)
3.2.6.6b	McNeese and Katz (1986)
3.2.6.7a	Woodson et al. (1992)
3.2.6.7b	Woodson et al. (1992)
3.2.6.8a	Woodson et al. (1992)
3.2.6.8b	Woodson et al. (1992); Vance (1987)
3.2.6.8c	Woodson et al. (1992)
3.2.7	Veron et al. (1990); Yeh and Silverstein (1989); Reinhart (1990); Beaton (1990); Snyder (1984); Williams (1990)
3.2.7.1	Beaton (1990)
3.2.7.2	Beaton (1990)
3.2.7.3	Yeh and Silverstein (1991)
3.2.7.4	Yeh and Silverstein (1991)

## REFERENCES (cont'd)

### Paragraph

### References

3.2.7.5	Yeh and Silverstein (1989); Yeh and Silverstein (1991)
3.2.7.6	Yeh and Silverstein (1991)
3.2.7.6	Yeh and Silverstein (1991)
3.2.7.7	Yeh and Silverstein (1991)
3.2.7.8	Yeh and Silverstein (1991)
3.2.7.9	Yeh and Silverstein (1991)
3.2.8	Dominessy (1989)
3.2.8.1	Baggen et al. (1988)
3.2.8.2	Baggen et al. (1988)
3.3	GSA (1991); 41 CFR, Chapter 201

**This page intentionally left blank.**

## 4.0 SCREEN DESIGN

Screen design refers to the way information is arranged and presented on a display screen. It is difficult to develop a complete set of standard screen design guidelines for the variety of DoD systems, primarily because of the diversity of tasks being performed by users. Screen design requirements can vary extensively, depending on the function being performed by the system. Some systems, such as information management systems that rely heavily on databases, do not usually require immediate user response to information displayed on their screens. On the other hand, real-time tactical command and control systems require the user to make immediate decisions and to input commands based on information on the display screen. Screen design requirements are unique for each system, depending on the system's primary function. The designer needs to understand the primary function of the system being developed to provide an effective screen design.

The designer should also incorporate the following general principles of Human Factors Engineering (HFE) design into the screen design, regardless of the system function:

- Guide the organization of information by Gestalt principles of perception, such as rules of:
  - **Proximity.** The human perception system tries to organize objects into groups if they are near each other in space.
  - **Similarity.** Objects are perceived as a group or set if they visually share common properties, such as size, color, orientation in space, or brightness.
  - **Closure.** The human visual perception system tries to complete the figure and establish meaningful wholes. The incomplete object or symbol is seen as complete or whole.
  - **Balance.** Humans prefer stability in the perceived visual environment. The presentation of materials at right angles and in vertical or horizontal groupings is easier to look at than curved or angled visual images.
- Design display formats to provide optimum transfer of information to the user by the use of information:
  - **Coding.** Coding is the assignment of meaning to an arbitrary visual cue. Examples of information coding include the use of color coding of friendly/hostile threat, editable/noneditable text fields, or shape coding of map symbols such as bridges/towns/roads/terrain, or font coding of mandatory/optional text fields, or combinations of these coding methods.
  - **Density.** Density is the percentage of character positions on the entire screen that contain data (Galitz 1993). It is recommended that screen data density not exceed 30 percent.

- **Grouping.** The general principle is that related information should be grouped together, but large groups of information should be broken up into subgroups. Related information is determined by what tasks are required to be performed by the user and by the users' perceptions of the information requirements.
- **Enumerating.** The presentation of information in numerical or alphabetic or chronological order.
- Present information simply and in a well-organized manner.
- Improve user performance by implementing the following screen features:
  - An orderly, clutter-free appearance
  - Information present in expected locations
  - Plain, simple language
  - A simple way to move through the system
  - A clear indication of interrelationships.
- Design display formats to group data items on the basis of some logical principle, considering trade-offs derived from task analysis.
- Design screens to minimize pointer and eye movement requirements within the overall design. The goal to minimize eye and pointer movement must be considered within general task considerations, with logical trade-offs taken into account.

The information presented in Subsections 4.2 and 4.3 represent design considerations that should be applied to the screen design of all DoD systems. The information presented in Subsection 4.1 can be applied to all DoD systems, but is primarily concerned with general security (GENSER) interfaces that are used to work with or display classified material.

## **4.1 INITIAL SCREEN DESIGN FOR ACCESS-CONTROLLED WORKSTATIONS**

This subsection provides guidelines for log-on, log-off, initial screen display, and management of access-controlled workstation resources. Although the focus is drawn from intelligence applications, the information presented applies to all system designs that display classified material and/or that control user access. The specific security requirements of the applicable domain relating to screen design must also be applied to any given application. This subsection applies primarily to GENSER requirements.

General principles that should be followed are that the system should provide both the necessary protection and be easy for the operator to use. The log-on for a system should not discourage use of the system by authorized users. Use of system resources and functions should be obvious and straightforward. Log-off should protect the data and preserve the information needed by the

user without complicated and time-consuming procedures. Details for HCI design of CMW are contained in Appendix A.

#### **4.1.1 Workstation Log-On**

Develop a standard workstation log-on screen for each system (see Figure 4-1). All workstations should implement a screen saver, rather than continuously display a log-on screen or other display on an idle workstation. When the workstation has been idle for a maximum of 5 minutes, a screen saver should be activated and deactivated when new activity is initiated. When appropriate, allow the user to set the screen saver activation time to less than 5 minutes.

Guidelines for developing a workstation log-on procedure are as follows:

- Ensure that security authentication information (when required) is a combination of name, password, and/or other identification information required before a user can access system resources.
- Ensure that each prompt for the user's name, password, etc. is clearly labeled and displayed on a separate line.
- Display error messages clearly on the computer screen along with guidance on how to correct the error. Error messages or help generated during the log-on sequence should not convey information that could assist someone in breaking into the system.
- When displaying a machine classification on a workstation accredited for system high operations, display the system high banner. The banner should be displayed in the color appropriate to the security level (see Paragraph 4.1.6).

#### **4.1.2 Application Log-On**

A primary DoD architectural objective for secure systems is to implement unitary log-on, but some systems will be unable to support this feature immediately. In systems where unitary log-on is not supported, many applications will require a separate authentication process before they can be accessed. Following selection of such an application by the user, display an additional log-on to prompt the user for the required authentication information.

#### **4.1.3 Application Log-Off**

Select the Exit function to accomplish application log-off. If work has not been saved, request that user confirm the quit, save modified data, or cancel the request. Application log-off exits an application and closes all windows associated with the application.

## LOG-IN SCREEN (Command Line System)

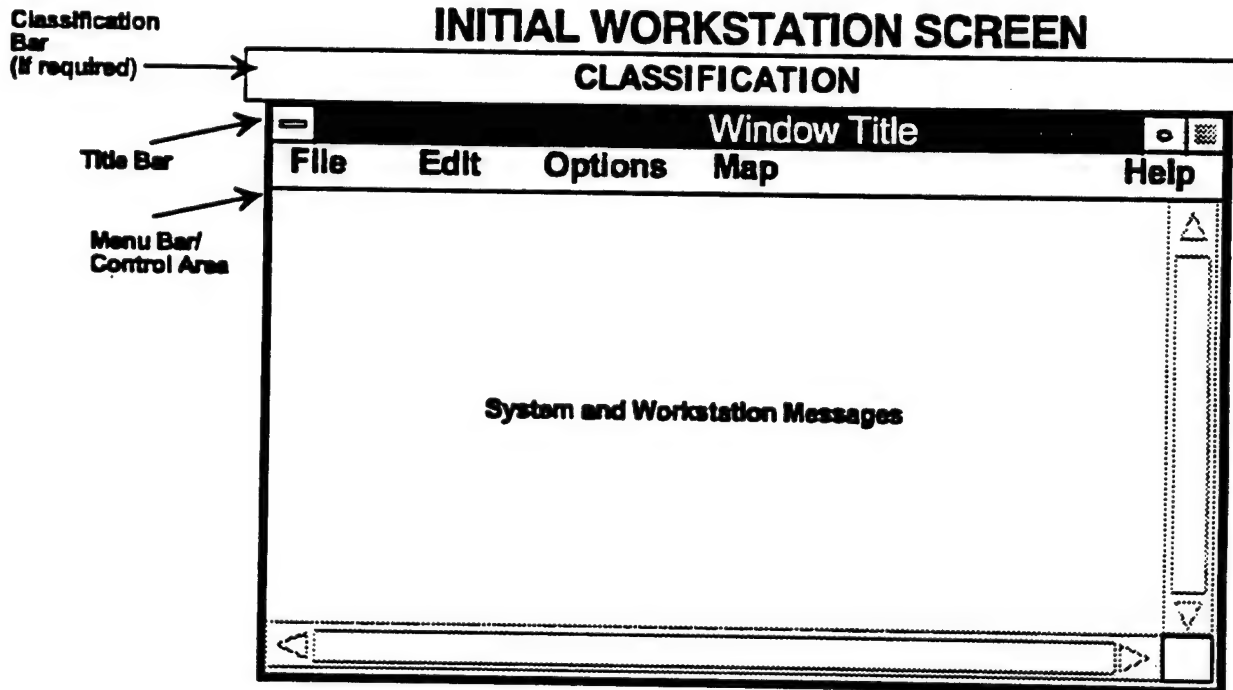
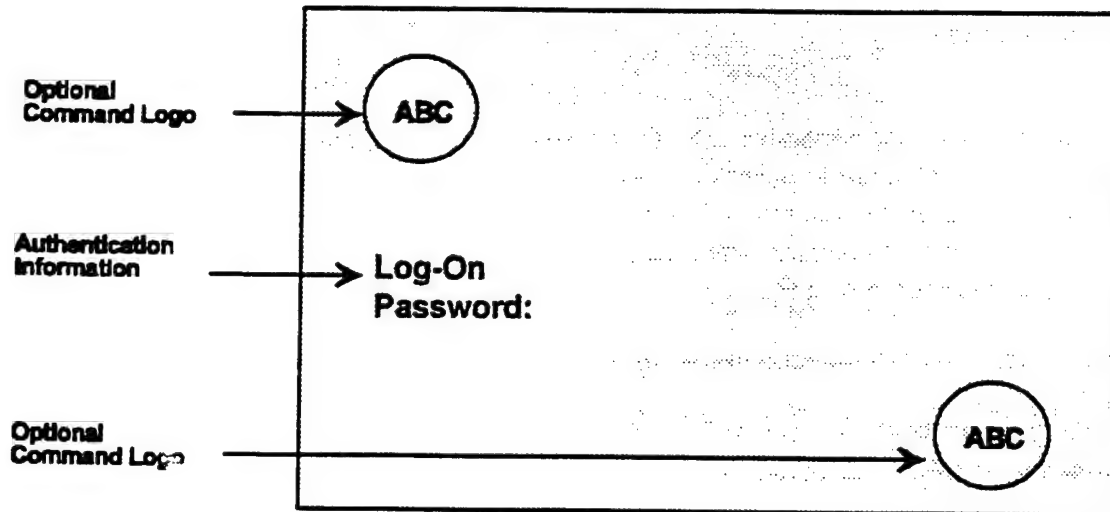


Figure 4-1. Sample Workstation Screens



#### **4.1.4 Workstation Log-Off**

Workstation log-off ends the session, closes all application windows, and returns the computer screen to the initial workstation log-on screen. If any applications are running, workstation log-off should also initiate an exit from all active applications. Workstation log-off requires user confirmation and is accomplished by selecting a log-off option from the resource manager window.

#### **4.1.5 Initial Workstation Screen Display**

When the user successfully completes the workstation log-on procedures and has been granted access to system resources, the initial screen should give the user access to the allowed system resources. Access can be accomplished by menus, icons, or interface structures, such as icon/tool bars. Some domains have specific requirements, such as the compartmentalized workstation requirement that a resource manager window will be displayed on the initial screen. Some tactical systems (especially sensor displays) will need to maximize available screen display space and will therefore design to minimize the space used by all other interfaces, including resource management interfaces. The designer of tactical systems may need to provide a resource management interface, such as an icon/tool bar with a toggle on/off option, to provide more user control of the screen display area. Resource management functions should be easily available to the user with a minimum of required keystrokes. The following are recommended basic resource management capabilities. General window functions are discussed in Section 5.0.

##### **4.1.5.1 Resource Management**

Resource management is the collection of functions that provides access to workstation resources and utilities (e.g., drives, printer, files, applications, software packages, etc.). This function is sometimes referred to as session management, but to avoid ambiguity, this *Style Guide* discusses the management of workstation resources under the generic term of resource management. The availability and display of resource capabilities should be determined by task requirements of the user. The following list of resource management capabilities provides examples of recommended functions:

- Program accesses
- Window snapshots (print screen)
- Access to common applications (e.g., word processor, spreadsheet)
- User preference/customization (e.g., left or right-handed mouse, color)
- Utilities (e.g., calculator, calendar, clock/alarm, note pad, mail)
- Display of system and workstation messages (error and status)

- End session/log user out of account
- Work file maintenance
- System-level help
- Security functions for authorized persons
- Device management capability (i.e., printer, mouse, facsimile [fax], etc.).

The resource management interface should present only those functions and applications a particular user is allowed to access. For example, only users authorized to perform certain security functions should have those options available within a resource management menu. Users may require data from several systems to perform their specific jobs. When multiple data sets must be accessed to satisfy a user query, it is the responsibility of the application to determine where the data reside.

#### **4.1.5.2 Resource Management Interface**

A resource management interface should contain, as a minimum, easy access for applications the user is authorized to use, including HELP. Workstations that require the display of system classification continuously can easily accommodate menus and/or icon/tool bar interfaces below the status display. Figure 4-1 illustrated a sample initial workstation screen. A long-term DoD objective is to implement user-oriented (e.g., help, messaging) resource management interfaces.

#### **4.1.6 Classification Color Selection**

The military intelligence community requires the colors listed below; the military community should follow these color selections. Classification bar color codes are shown as follows with their associated meanings.

<b>Bar Color</b>	<b>Meaning</b>
Green	Unclassified
Blue	Confidential
Red	Secret
Orange	Top Secret
Yellow	Sensitive Compartmented Information

When Sensitive Compartmented Information is displayed, both the classification (Secret or Top Secret) and "Sensitive Compartmented Information" must be displayed in the classification bar. The classification bar should contain two colors, orange or red, as appropriate, and yellow. If a

two-color classification bar is unfeasible, a yellow classification bar should be displayed in which the classification and "Sensitive Compartmented Information" is displayed.

## **4.2 SCREEN DESIGN GUIDELINES**

The visual design of the interface has increased in importance with the broad adoption of GUI as a standard interface. The screen design should be pleasing to the user, with screen elements arranged to be visually, conceptually, and linguistically clear and understandable. The visual presentation should be compatible with user expectations, using familiar concepts, terminology, and work flow. The concept of an easily learned and understood interface is central to the creation of screen designs. The interface should have the flexibility to support the requirements of users ranging from novice to expert. Users will be more productive if they control the interaction with the application, and this approach is recommended where possible. The best designs are easy to configure and reconfigure. One reason the use of prototypes for user input and feedback is recommended is because it will enhance the visual design process.

The functional design of the screen must deal with compatibility between the design and tasks to be performed. The user must be able to perform required tasks in a direct and obvious manner. Task flow should be predictable by the user and easy to follow. The functional layout should be efficient for the user, minimizing keystrokes and hand/eye movements where possible without impacting functional effectiveness. The functional interface should deal with errors and mistakes in a manner that allows easy correction and recovery. Responsiveness of the interface is important to screen design, together with consistency of the design. The most effective designs allow the user to rely on an interface that remains consistent throughout all system screens. The basic principle of functional screen design is to keep the interface as simple as possible and provide all the functionality required by the user to do the job.

The interface must conform to standards, guidelines, and requirements. The domain to which the system belongs, the functional system requirements, and the hardware platform selected will define many of the standards, guidelines, and requirements for the system. The screen interface should make hardware system-specific operations as transparent to the user as possible. The designer should review the standards, guidelines, and requirements before starting the design process. The design process should allow for trade-off in design to accommodate user needs and provide a means for user feedback into the design trade-off process.

### **4.2.1 Visual Design**

The screen should be visually pleasing to the user. Using size, shape, location, and color can be counterproductive if these features startle or surprise the user. The goal of good visual design is computer interface design that visually encourages work flow, is easy to look at and easy to use.

#### **4.2.1.1 Consistent Display Structure**

Create display formats with a consistent structure evident to the user, so that display features are always presented in the same way.

#### 4.2.1.2 Consistent Fields

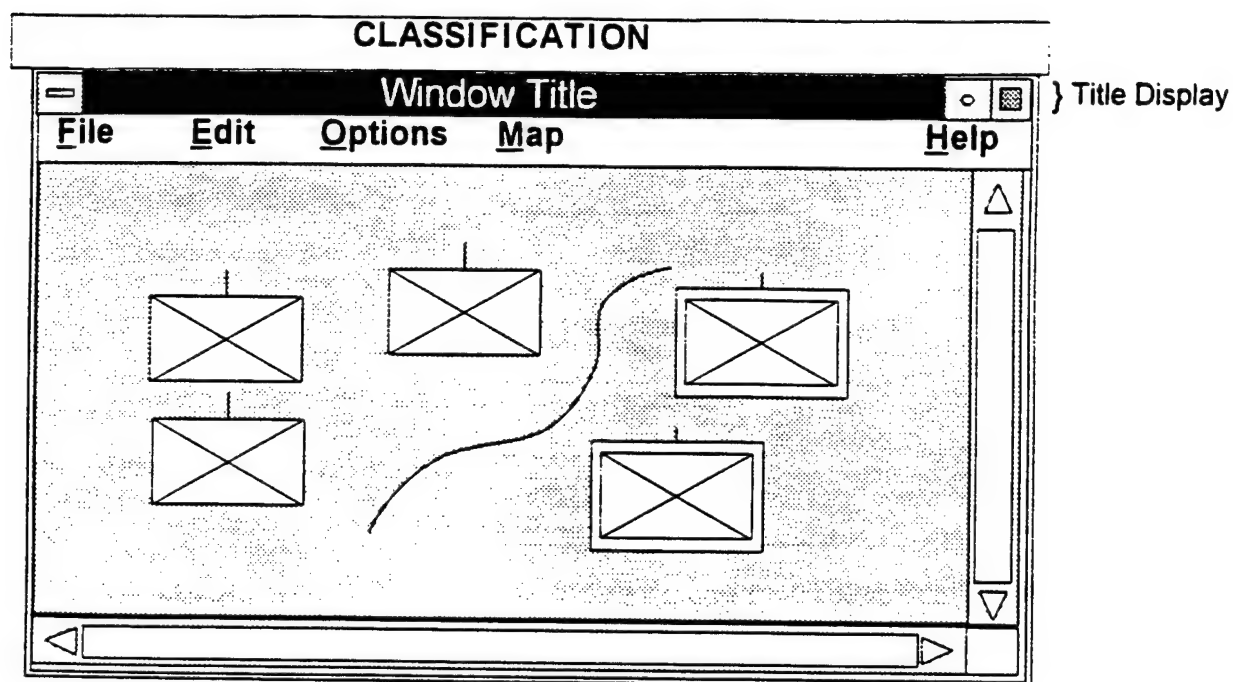
Use fields, such as data element names, group captions/titles, or window titles, that are consistently located and remain the same for each presentation.

#### 4.2.1.3 General Format

- Make the different elements in a display format distinctive.
- Organize information on a display screen such that visual competition among distinct items of information is minimized.
- Use contrasting features, such as inverse video and color, to call attention to different screen components and urgent items.
- Arrange screen elements to be visually, conceptually, and linguistically clear and understandable.

#### 4.2.1.4 Screen Organization

- Ensure that the order of data follows some principle that can be recognized and applied by the user.
- Begin every display with a title or header located at the top of the page or window, briefly describing display contents or purpose, as in Figure 4-1. In the special case of a CWS, a security banner must be the top-level label.
- Ensure that the area set aside for displaying messages is consistent. Text systems typically reserve the last few lines at the bottom of displays for status and error messages, prompts, and command entry, when appropriate (see Figure 4-2). This area is also used for a supporting data menu bar, including such items as a user note pad.
- For text displays, ensure that screen or focus window density (i.e., ratio of characters to blank spaces) does not exceed 60 percent of available character spaces. The data or information density (i.e., ratio of data characters to total display space) should not exceed 30 percent of the total screen or window. In this case, window size is defined as the default display size, not the maximum or minimum window size. For example:
  - A focus window 20 characters wide and 5 lines high has a total character space of 100 characters. The following sentence: *"The quick brown fox jumped over the large black dog."* uses 43 character spaces or 43 percent of the available 5-line display. If the following sentence is added: *"The information can be overloaded fast!"*, the total character space used is 77 characters or 77 percent of the total space. Thus, adding the second sentence creates too high a density for the size of the focus window.

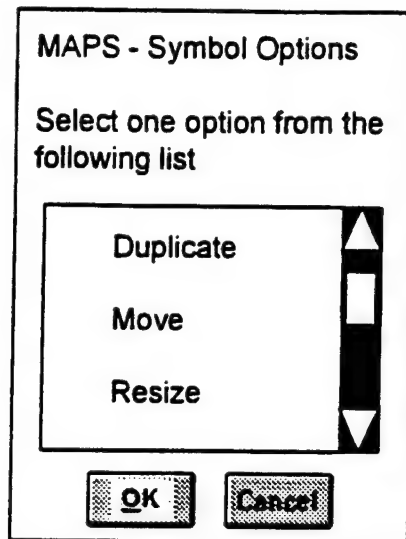


OR

Class III Item Status		} Title Display
Item	Number on Hand	
F46 Gas CMST 92	50	
F54 Diesel Fuel	200	
F40 Jet Fuel JF4	175	

**Figure 4-2. Example of How Specific Types of Information Should Be Located on a Window**

- Calculating density is based on improving readability of displayed text. The display area does not include controls that may be within the window border. Controls are normally located in an area that is not made available to display the textual data.
- The principle of screen density applies to the display of graphic figures, but no standards for density maximums for graphic displays are available. The developer would be advised to test/prototype screen density designs with the user community.
- Highlight the instructions on how to use a screen or window at the top of the text, preceding response options, as illustrated in Figure 4-3. Include instructions on the disposition of a completed screen or window at the bottom.



**Figure 4-3. Example of the Proper Location of Display Screen Instructions**

- Assign functional fields for particular kinds of data, such as program messages, error messages, system messages, and alarms. These fields and displays should be consistently located in the physical space of the screen or window.

#### **4.2.1.5 Primary Viewing Area**

When data and terms are particularly important, require immediate user response or, when they are more frequently displayed, group them in the primary viewing area of the user.

#### **4.2.1.6 Arrangement of Data on Screen**

Arrange and group data on application display screens to differentiate between instructions and data and to facilitate observation of similarities, differences, and trends for the most common uses.

#### **4.2.1.7 Cohesive Groupings**

Provide cohesive groupings of screen elements by using blank space, surrounding lines, different intensity levels, etc.

#### **4.2.1.8 User Attention**

Techniques that direct user attention should be used carefully or they will lose their effectiveness. A number of visual techniques may be used to attract the attention of the user.

The following list contains visual approaches, their properties and parameters that may be used to attract user attention:

- **Intensity:** Do not use more than two levels.
- **Marking:** Underline, arrows, bullet, dash, asterisk.
- **Size:** The maximum number of sizes should be four or less.
- **Blinking:** Blink rates should be in the 2 to 4 hertz range.
- **Choice of Fonts:** The number of fonts should be three or less.
- **Inverse Video:** Use inverse coloring.
- **Color:** Use up to four standard colors.

#### **4.2.1.9 User Feedback (Prototyping)**

The visual screen design process should include prototype or sample screen presentations to users. Sampling user opinion and obtaining user input to the visual design process is critical to developing an effective interface for a system. The basic principles of screen design require that user feedback be an integral part of the development process.

#### **4.2.2 Functional Screen Design**

The functional screen design integrates the user task requirements with the available computer functionality to optimize the user's performance. The interface must present an obvious and predictable work flow that is efficient for the user. The interface should be as simple as possible and still give the user the functionality to complete tasks effectively.

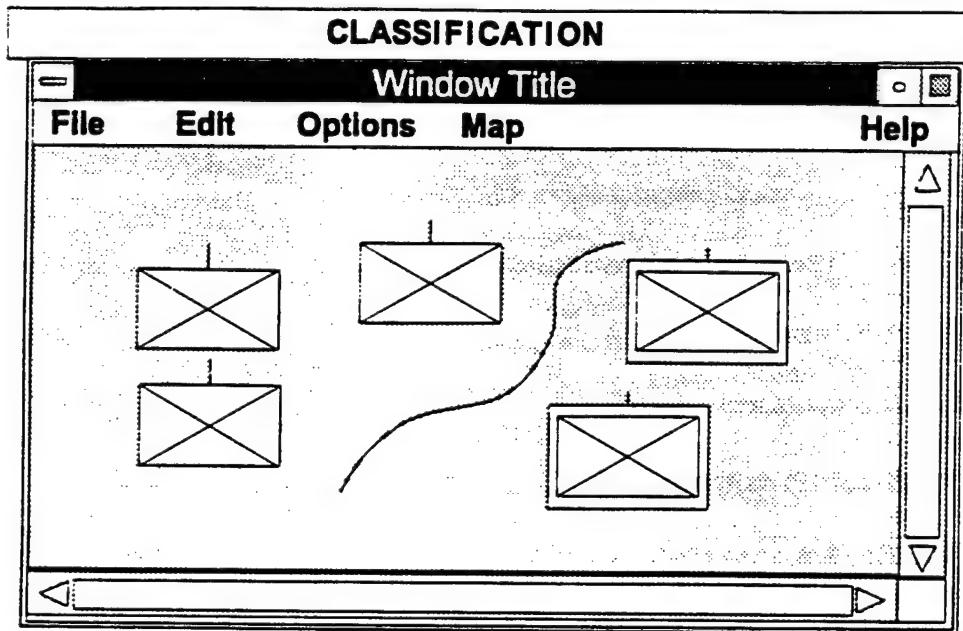
##### **4.2.2.1 Information Display Based on Criticality**

The screen design procedures for a system should establish a set of criteria for prioritizing different levels of displayed information. For example, in military tactical systems, critical tactical information should always be displayed, whereas optional information should be available by request. See Figure 4-4 for an illustration of this principle, using a military land-based situation map overlay. The position of the units is critical information, whereas details of the unit are available on a pop-up box.

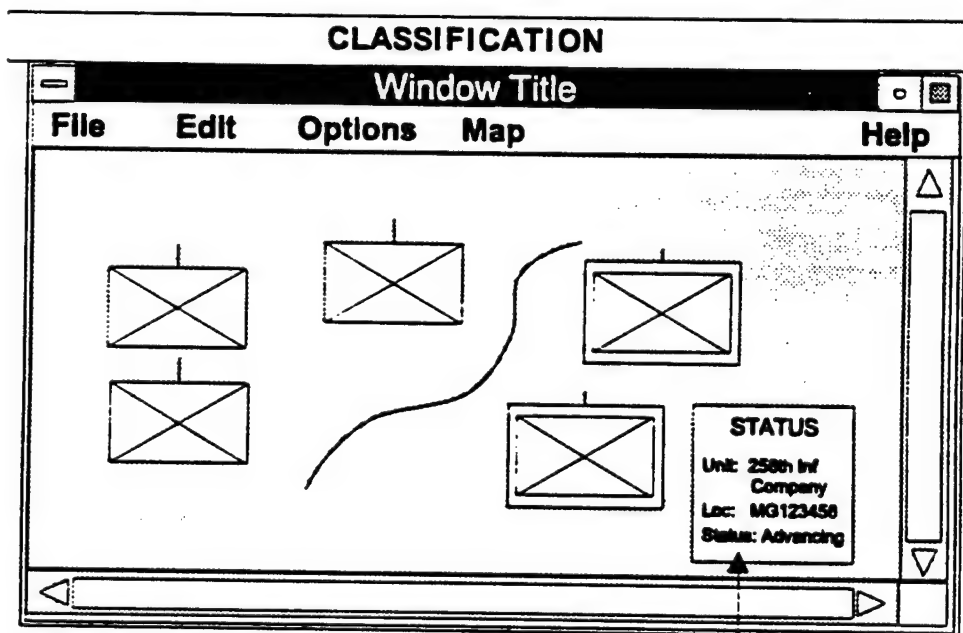
##### **4.2.2.2 Display Only Critical Information**

Minimize the text information density on a system display by presenting only information essential to the user at the time. High density of information on a screen or window results in the user needing extra time and making errors in finding information. It is recommended that screen or window information density be less than 30 percent. Information is variable data and should be distinguished from labels, titles, or lines/boxes. The density of graphic displays should also be minimized, using input from end users as guidance.

Design Where Critical Information is Always Present



Optional Information Added to Critical Information



Optional

Figure 4-4. Tactical Information Display Options



#### **4.2.2.3 Integrated Display**

When the user needs specific data displayed concurrently to judge a time-critical task (e.g., a military tactical situation), provide those data in an integrated display rather than partitioning them into separate windows. Using integrated data displays will facilitate decision-making and time-constrained tasks.

#### **4.2.2.4 Information Format**

Present information in a directly usable form. Do not require the user to decode or interpret data. The structure of information presented on the screen should be consistent. This helps the user develop a perceptual model of the interface.

#### **4.2.2.5 Grouping for Data Comparison**

If users must analyze sets of data to discern similarities, differences, trends, and relationships, structure display formats so the data are grouped consistently.

#### **4.2.2.6 Important Data Placement**

Where displayed data are used in some spatial or temporal order, consider grouping those data by sequence of use to preserve that order.

#### **4.2.2.7 Efficient Layout**

A design goal for DoD is to minimize keystrokes and hand/eye movements, but this goal must be implemented carefully. The designer should consider functional requirements and trade-offs among task requirements, while striving for the goal of minimum keystrokes and hand/eye movements. An efficient layout will incorporate consideration for function as well as interaction.

#### **4.2.2.8 Error Management**

The screen design must include error management. The most effective error management is an interface design that minimizes errors. Errors, in fact, will occur, and the functional interface must include provision for managing errors and mistakes related to functional tasks performed by the user. The interface should allow easy correction and recovery while protecting the application from catastrophic user errors. See Subsection 8.2 ON-LINE HELP for more information.

#### **4.2.2.9 Functional Trade-Off**

A functional trade-off analysis for the system under development should be performed to determine the tasks best performed by automation and the tasks best performed by humans.

Automation can be an efficient partner by handling routine tasks, while reducing the impact of tedious and error-prone tasks. The functional screen design must assign the task to the most

appropriate resource, either computer or human. Tasks should not be automated for the sake of automation, but the user should not be burdened with tasks better done by automation. See the list below, which is based on a table from Shneiderman (1992, page 84).

#### **Tasks Best Performed By Humans**

- Remember principles and strategies
- Retrieve pertinent details without a prior connection
- Adaptability
- Reason inductively - generalize from observations
- Sense unusual and unexpected events
- Act in unanticipated emergencies and novel situations
- Draw on experience and adapt decisions to situation
- Detect stimuli in noisy background

#### **Tasks Best Performed By Automation**

- Recall quantities of detailed information
- Process quantitative data in prespecified ways
- Accuracy
- Reason deductively - infer from a general principle
- Monitor prespecified events
- Perform repetitive preprogrammed actions reliably
- Perform several activities simultaneously
- Calculate accurately and quickly

### **4.2.3 Screen Design Standards, Guidelines, and Requirements**

#### **4.2.3.1 Format**

- Use abbreviations appropriately and consistently. Provide a key or built-in reference table. Abbreviations should conform to standards (e.g., AR310-50 [U.S. Department of the Army 1985a], MIL-STD-12D [DoD 1981], MIL-STD-411E [DoD 1991], and MIL-STD-783D [DoD 1984]). The domain-level style guide should cite the domain-selected standard for abbreviations. Do not place periods after abbreviations. Applications requiring extensive text input should provide an on-line spell-checker that addresses abbreviations and acronyms.
- Use short, simple statements in text.

#### **4.2.3.2 Data Organization**

- Break large portions of text into smaller, meaningful groups to minimize the amount of information to be attended to at one time. See the examples below.

#### ***Poor:***

*The 3rd Bn is currently located at 32UNA100100, moving to contact in sector 8, with 80% strength, supported by an armor platoon.*

**Good:**

**3 Bn Status:**

- *At 32UNA100100*
  - *Moving to Contact in Sector 8*
  - *80% Strength*
  - *Supported by an armor platoon*
- Use blank space to structure a display.
  - For screens containing large amounts of text, consider using two columns of text to improve readability.
  - Ensure labels are sufficiently close to their related data fields but separated by at least one space.
  - Provide adequate spacing between words and lines of text for better legibility. Separate paragraphs with a blank line.
  - Present a series of data elements vertically, not horizontally, in text, as follows.

#### **Vertical - Easy to Read**

<b>Class I</b>	<b>Class II</b>	<b>Class III</b>
Item 1	Item 1	Item 1
Item 2	Item 2	Item 2
Item 3	Item 3	Item 3

#### **Horizontal - Difficult to Read**

<b>Class I</b>	<b>Item 1</b>	<b>Item 2</b>	<b>Item 3</b>
<b>Class II</b>	<b>Item 1</b>	<b>Item 2</b>	<b>Item 3</b>
<b>Class III</b>	<b>Item 1</b>	<b>Item 2</b>	<b>Item 3</b>

- Provide an obvious starting point for information.
- Justify columns, as noted and illustrated below.
  - Left-justify alphanumeric columns to permit rapid scanning.
  - Right-justify numerical data without decimals.
  - Justify numerical data with decimal points by the decimal.

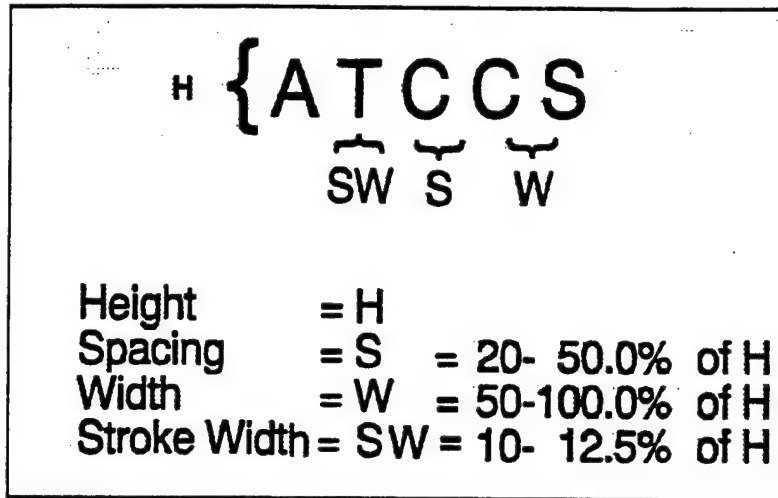
Poor	Good
Washington DC	Washington DC
Cars	Cars
People	People
400	400
4210	4210
39111	39111
1.5	1.5
10.36	10.36
1.365	1.365

#### 4.2.3.3 Line Organization

- On a full-screen text display, use only 70 character positions on the standard 80-character line to increase reading efficiency. This is most important when detailed reading is the user's primary task. When displaying text in windows format, where possible, ensure word wrap to prevent excessive scrolling side-to-side. Text presented in a window should have blank space at the start and end of each line to increase readability of the text. It is recommended that the blank space be approximately the size of two average text widths.
- Display no more than 35 to 40 characters per column on each line for information presented in columns.

#### 4.2.3.4 Character Design

- Use capital letters for typographic coding, headlines, and where special emphasis is required, such as some captions and labels. However, it is usually best to capitalize only the first letter, for example, in a horizontal series such as button labels.
- Do not use all capital letters in running text or tables, as this impairs word recognition, reduces readability, and limits space between text lines.
- Ensure spacing between characters (both fixed and proportional width) at 20 to 50 percent of character height. Spacing between lines should be equal to character height (see Figure 4-5).



**Figure 4-5. Example of a Character Size and Spacing**

- Ensure that the minimum height of displayed characters is 1/200 of viewing distance (approximately 17 minutes of visual angle). Therefore, a viewing distance of 36 inches requires 0.18-inch character height on the display screen. Character width should be 50 to 100 percent of character height. Character stroke width minimum is 10 to 12.5 percent of character height. Maximum text size should not exceed 10 percent of the available vertical display area on a full-size screen.
- Ensure that characters contain a minimum 7 x 9 dot matrix construction for better readability
- Some applications require over-the-shoulder reading of characters on the screen and should be legible to a person standing behind the user (e.g., operations in a tactical environment). Ensure that the screen viewing distance referred to in Paragraph 4.2.3.4d reflects the anticipated maximum viewing distance. Large fonts with broad stroke widths are recommended to improve readability. Selections of background color and contrasting foreground (text) color should ensure sufficient contrast for good readability.
- The usual font size designation is given in points. Display of text fonts on screens is proportional to point size, but the actual size of displayed text is related to screen size and application software. Font point size only controls the actual size of printed output. It is recommended that screen text size be reviewed and adjusted in relation to the objective hardware system.

#### **4.2.3.5 Scrolling of Data**

- Display text information statically on the screen, rather than constantly scrolling it across the screen.

- If text is meant to be scanned by constant scrolling, columns with 35 characters or fewer per line are preferred.

#### **4.2.3.6 Data Grouped Alphabetically or Chronologically**

When no appropriate logic exists for grouping data by sequence, function, frequency, or importance, adopt some other principle, such as alphabetical or chronological grouping.

#### **4.2.3.7 Paging Crowded Displays**

When a window contains too much data for presentation in a single pane, normally the window has the ability to scroll the data. It is recommended that side-to-side scrolling be avoided where possible. If the data must be presented in block or groups, partition data into separately displayable window panes or pages. Refer to Section 5.0 WINDOWS.

#### **4.2.3.8 Related Data on Same Page**

When partitioning displays into multiple pages, take into account the type of data being partitioned, and display functionally related data items together on one page.

#### **4.2.3.9 Multiple Pages Labeling**

In a multipage display, label each page with a unique identifier that shows its relation to the other pages (e.g., page 1 of 5).

#### **4.2.3.10 Screen Viewing Distance**

The minimum viewing distance from the user's eye to the monitor must be equal to or greater than 30 centimeters (cm) or 12 inches. For best visual acuity, it is recommended that the viewing distance be set to the range of 30 to 40 cm (12 to 16 inches). Viewing distance must be considered when character and symbol sizes are determined for the screen design.

#### **4.2.3.11 Text Readability**

The readability of displayed text is maximized if the character height is in the range of 16 to 24 minutes of visual arc (min). The preferred size is 20 to 22 minutes of visual arc. The designer must determine the maximum viewing distance from the display, then calculate the minimum size of the text, using the formula:

$$\text{Visual Angle (min)} = \frac{(57.3) (60)L}{D}$$

where L = size of the object, and D = distance from the eye to the object.

The list on the following page provides examples of calculations based on the above formula. For example, to obtain a visual angle of 20°, when the user is 32 cm from the screen, this requires a screen object (font height) to be 0.186 cm.

#### 4.2.3.12 Input Prompts

When command line interfaces are used, display the input prompt at a standard location, next to the command entry area of the display.

### 4.3 COLOR

The use of color as an information discriminator is crucial, especially as emerging command and control systems implement GUIs and high resolution color graphics displays. Color can be a very effective discriminator, for example, by decluttering a display and thus improving task performance. Color can also introduce the very clutter and performance degradation it attempts to reduce. For these reasons, color in a display must be used very carefully.

The individual or team responsible for screen design must be sensitive to the many factors that affect how a person perceives and reacts to color as an information discriminator. An in-depth discussion on visual perception, color, and human performance is beyond the scope of this document; nevertheless, basic information is needed. Definitions of key terms associated with this subject are provided.

Visual Angle	D = Distance to object (cm)	L = Height of object (cm)
17 min	30 cm	0.148 cm
17 min	31 cm	0.153 cm
17 min	32 cm	0.158 cm
17 min	33 cm	0.163 cm
17 min	34 cm	0.168 cm
18 min	30 cm	0.157 cm
18 min	31 cm	0.162 cm
18 min	32 cm	0.167 cm
18 min	33 cm	0.172 cm
18 min	34 cm	0.178 cm
19 min	30 cm	0.165 cm
19 min	31 cm	0.171 cm
19 min	32 cm	0.177 cm
19 min	33 cm	0.182 cm
19 min	34 cm	0.188 cm
20 min	30 cm	0.174 cm
20 min	31 cm	0.180 cm
20 min	32 cm	0.186 cm
20 min	33 cm	0.192 cm
20 min	34 cm	0.198 cm

## Definitions of Key Terms for Color Usage

TERM	DEFINITION
Achromatic	Colorless; lights that have no definite hue, black and white are achromatic.
Brightness	The perceptual (psychological) correlate of intensity that ranges from dark to bright.
Chroma	The intensity or vividness of a color, its brightness or dullness.
Chromatic	Highly colored.
Discrimination	Degree to which a human visual system can sense differences in the physical characteristics of an image.
Hue	The psychological attribute of color sensation associated with the physical property of visible wavelengths. The name of a color such as blue, red, green, or orange.
Legibility	Ability to identify an alphanumeric character or symbol. A criterion of image quality.
Luminance	The amount of light reflected or emitted by a surface, measured in footLamberts.
Luminance	Ratio of the foreground brightness compared to the Contrast background brightness.
Monochromatic	Consisting of one color or hue.
Recognition	Ability to recognize or interpret the meaning or association of an image.
Saturation	The degree to which a hue differs from a gray of the same lightness.
Shade	The darkness of a hue produced by adding black.
Tint	The lightness of a hue produced by adding white.

The designer should recognize the following important guidelines for using color in computer display systems:



- Both brightness and type of lighting (e.g., incandescent versus fluorescent) can affect how colors are perceived. For example, bright ambient light desaturates display colors, leading to degraded color identification and discrimination. It may shift the eye's adaption, also reducing the ability to discriminate color. In essence, identically colored objects can be perceived as being dissimilar under different lighting conditions.
- How the color of an object is perceived is directly related to the color surrounding it.
- Visibility and readability are a direct result of the contrast between the foreground and the background.
- Use color sparingly as an information discriminator. Color rapidly loses meaning and, when overused, may impede rather than enhance human performance.
- Use colors consistently within a display and across a set of displays for an application.
- Ensure that the meaning of color is consistent with user expectation.
- When using color to impart a specific meaning to the user, use an additional, redundant form of coding, such as pattern or shape. This ensures that the correct meaning will be conveyed should the user have a color vision deficit, or should color be unavailable on the screen.
- Standardize color coding for operational applications. Although flexibility in color coding schemes may be desirable for a terminal dedicated to a single user, color coding should be standardized for operational applications. Because of the variety of users on a tactical terminal, a terminal with a uniquely customized color coding scheme may be very difficult to interpret. Allow the user to select color schemes for aspects of the application that do not involve coding or status, unless the primary task performed by the user requires the capability to manipulate these colors and making these changes will not negatively impact the performance of other users. A default scheme should be easily available to restore the interface for subsequent users.
- A requirement for adjustable colors is created because of portable applications among hardware configurations. Each hardware system display has different color perceptual values and color names. Portable applications must accommodate these differences. Assign status colors during installation; allow the user to adjust these colors only if essential to the task being performed with the system.

The following subsections provide more detailed guidelines for using color in DoD systems. Note that many guidelines contained in this subsection are based on the use of color in text-based software, primarily because the majority of past research in color usage was done with text applications. Although results of research with GUIs and graphical presentation tend to confirm these principles and new information is emerging, more research is needed on using color in tactical graphics applications, especially in foreground/background combinations for colored map graphic displays.

The designer should also note that it is easier to define color combinations to avoid than to identify a single best way to utilize color. Color choice and/or color combination tend to be a matter of personal preference. For example, high contrast between foreground text and background is crucial but can be accomplished with a number of color combinations. This ambiguity becomes all the greater when developing design guidance for the different command and control applications represented across military systems. The designer should utilize those domain-level guidelines most relevant to the particular application.

Use color when a basic monochromatic presentation of tactical information needs to be augmented for the user to gain a more effective understanding of the information being presented.

#### **4.3.1 General**

Reference to color can mean a family of color, such as reds; it can mean a Red, Green, Blue (RGB) definition, such as red 1,0,0; or it can mean a specific frequency of light. The *Style Guide* will use an underline for color names that refer to primary (i.e., full gun) RGB. Otherwise the reference to color is to a family or perceived color.

##### **4.3.1.1 When to Use**

Use color carefully (as a coding method, color can rapidly lose its effectiveness). When necessary, use color:

- To attach specific meaning to tactical information presented in the form of text or symbology
- To direct user's attention to the most important or time-critical information on the screen (i.e., information category headings, system and user errors, information requiring immediate attention, key data items, window titles)
- To enable a user to differentiate rapidly among several types of information, especially when the information is dispersed on the display
- To increase the amount of information portrayed on a graphic display by adding color in addition to shape
- To indicate changes in the status of graphical data.

##### **4.3.1.2 Constraints on Use**

The user with defective color vision will have difficulty discriminating among the colors. Color vision deficiency occurs in about 8 to 10 percent of the general male population and about 0.4 to 0.6 percent of the female population in the United States. Consider the following when including color in display screens:

- Add color coding only after displays have already been designed as effectively as possible in an achromatic format.
- Color only logically related information with similar hues. Consider spacing or highlighting instead of, or in addition to, color.
- When emphasizing tactical information by means of color, use a color for more important information that is brighter than adjacent color-coded information. Ensure the choice of colors is consistent with the user's expectations for the information being coded (see Paragraph 4.3.2).
- Do not use color coding when it might confuse users with defective color vision or when the use of color reduces screen readability. If color must be used, consider the following:
  - When the user must compare data, such as those contained in graphs based on color, the list below suggests combinations to use and avoid as comparison colors for application information requiring important or frequent discriminations. These combinations do not apply to text (foreground) and screen (background) color combinations.

#### **Data Comparison Color Combinations**

##### **RECOMMENDED**

White/Green  
 Gold/Cyan  
 Gold/Green  
 Green/Magenta  
 Green/Lavender  
 Cyan/Red  
 White/Gold/Green  
 White/Gold/Blue  
 White/Gold/Magenta  
 White/Red/Cyan  
 Red/Cyan/Gold  
 Cyan/Yellow/Lavender  
 Gold/Magenta/Blue  
 Gold/Lavender/Green

##### **AVOID**

Red/Blue  
 Red/Green  
 Red/Purple  
 Red/Yellow  
 Red/Magenta  
 White/Cyan  
 White/Yellow  
 Blue/Green  
 Blue/Purple  
 Green/Cyan  
 Cyan/Lavender  
 Red/Yellow/Green  
 Red/Blue/Green  
 Red/Magenta/Blue  
 White/Cyan/Yellow  
 Green/Cyan/Blue

- Especially when using color combinations from the "avoid" side of the preceding list, provide additional cues, such as brightness and saturation, to enhance discriminability.
- Do not code solely by color. Make color coding redundant with some other display feature, such as shape, pattern, or actual text content.
- Avoid requiring the user to discriminate between colors in small areas of the display. Small, color-coded areas are subject to loss and bleeding of colors. Use achromatic colors (i.e., black or white) if coding must be done in small areas.
- White is a good choice to highlight data that require particular attention, except on a display with white or near-white background. Do not use white excessively as a highlighter, as it can create a glaring brightness that may interfere with screen legibility. When status changes are signaled by color, do not use that color to highlight text. Signal any status changes using color coding by a ball or box next to the text.
- Ensure that contrast is high between the text or graphical object and its background, to enhance screen readability. Generally, the color foreground should differ from its background by a minimum of 100 E (Commission International d'Eclairage (CIE) Yu'v') distances, when luminance values range from 0 to 255. A luminance equation (\*intensities have been normalized from 0.0 to 1.0) that can be used for contrast determination is:

$$Y = .30 * \text{Red} + .59 * \text{Green} + .11 * \text{Blue}$$

The minimum normalized difference should be greater than 0.35 for good contrast. The luminance difference value for a number of standard colors is listed below.

- Based upon the tables listed, values **red and black** or **black and blue** should not be used, while **white and blue** or **black and cyan** have acceptable contrasts. Minimum luminance contrast ratios are required for specific tasks. For discrimination and legibility, acceptable ratios of foreground-to-background luminance contrast range from 6:1 to 10:1, or 1:6 to 1:10. The list below provides guidance for specific conditions. Using pure white or black as a background color is not recommended. Unsaturated hues provide the best background contrast. The use of high luminance backgrounds tends to cause user eye strain. Therefore, task requirements should be considered when selecting high or low background illumination.

**NOTE:** *When calculating luminance ratios using black (i.e., zero luminance), use a value of one (1) to avoid dividing by zero (0). This is based on luminance values that range from 0 to 255.*

### Normalized Standard Color Luminance (Y)

	<b>R</b>	<b>G</b>	<b>B</b>	<b>Y</b>
Black	0.0	0.0	0.0	0.00
White	1.0	1.0	1.0	1.00
Red	1.0	0.0	0.0	0.30
Green	0.0	1.0	0.0	0.59
Blue	0.0	0.0	1.0	0.11
Cyan	0.0	1.0	1.0	0.70
Magenta	1.0	0.0	1.0	0.41
Orange	1.0	0.5	0.0	0.60
Yellow	1.0	1.0	0.0	0.89

### Luminance Differences

	<b>Black</b>	<b>White</b>	<b>Red</b>	<b>Green</b>	<b>Blue</b>	<b>Cyan</b>	<b>Magenta</b>	<b>Orange</b>	<b>Yellow</b>
Black	XXX	1.00	0.30	0.59	0.11	0.70	0.41	0.60	0.89
White	—	XXX	0.70	0.41	0.89	0.30	0.59	0.41	0.11
Red	—	—	XXX	0.29	0.19	0.40	0.11	0.30	0.59
Green	—	—	—	XXX	0.48	0.11	0.18	0.01	0.30
Blue	—	—	—	—	XXX	0.59	0.30	0.49	0.78
Cyan	—	—	—	—	—	XXX	0.29	0.11	0.19
Magenta	—	—	—	—	—	—	XXX	0.19	0.48
Orange	—	—	—	—	—	—	—	XXX	0.30

## **Recommended Luminance Contrast Ratios**

<b>CONDITION</b>	<b>RATIO OF FOREGROUND TO BACKGROUND</b>
Bright Ambient Illumination	>7:1
To Attract Attention	>7:1
To Sharpen Edges	>7:1
Continuous Reading	3:1 to 5:1
Dark Ambient Illumination	3:1 to 5:1
Camouflage Images or Smooth Edges	<3:1

### **4.3.2 Color Selection**

#### **4.3.2.1 General**

- When selecting colors for coding discrete categories of information displayed on a screen, ensure that those colors are easily discriminated in all expected operational environments.
- To aid in color discrimination, use colors that are as widely spaced along the visible color spectrum as possible. The following colors, listed by their wavelengths in millimicrons, are spaced widely enough for easy discrimination from one another.

#### **Color Wavelengths in Millimicrons:**

Red	700
Orange	600
Yellow	570
Yellow-green	535
Green	500
Blue-green	493
Blue	470

- Use an unobtrusive color to display information used infrequently on a screen. Unobtrusive colors have shorter wavelengths.
- Use warm colors (colors with a longer wavelength, such as red or orange) to convey action or the requirement for a response. Use cool colors (colors with a shorter wavelength, such as blue or green) to convey status or background information.
- Ensure that each color represents only one category of displayed data (i.e., those defined in Paragraph 4.3.2.6). A mismatch of color and color association slows recognition time and increases misidentification of words.

#### 4.3.2.2 Consistency

- Apply color consistently from screen to screen and from application to application to ensure that the user can make the proper interpretations. This is applicable both within and across DoD systems. For example, do not use status colors as window borders unless status coding is intended.
- Color coding should be consistent with the interaction of the label's color and the color associations of the words in the label. For example, the word ENEMY, if color-coded, should be red rather than green.
- Choose colors for coding based on conventional associations with particular colors. These should conform, if possible, to those specified in the appropriate domain-level documents, such as *Army FM 101-5-1: Operational Terms and Symbols* (U.S. Department of the Army 1985b).

#### 4.3.2.3 Number of Colors to Use

- Implement color coding conservatively, using relatively few colors to designate critical categories of displayed data and only where it will help user performance.
- Use no more than four colors at one time when using alphanumeric screens, with a maximum of seven total for all screens.
- Use four standard colors, reserving others for occasional use. Humans can easily discriminate only eight or nine highly saturated colors; the recommendation is *not* to exceed seven. Extensive coloring creates a brighter-than-necessary display, with subsequent negative impact on user performance.

#### 4.3.2.4 Pairing of Colors

Colors should be carefully paired on a screen to maximize human performance.

- Avoid simultaneous use or close proximity of highly saturated, spectrally extreme color pairs on a display screen. Examples include such color pairs as red and blue, yellow and purple, or magenta and green. This creates an effect where one colored object will appear closer than another (a 3D effect called chromostereopsia). This effect is most significant with red and blue.
- To emphasize different tactical information in text and presentation graphics displays, the color choice rules may be broken and contrasting colors such as red and green or blue and yellow may be used. However, in color choice, be consistent with the guidance provided in other parts of this section.
- To convey similarity in tactical information in text and presentation graphics displays, use similar colors, such as orange and yellow or blue and violet.
- Avoid using extensive coloring (e.g., many different colors) for the background, segments of the background, or particular regions surrounding individual characters or symbols.
- Avoid using pairs of monosaturated primary colors, such as red-green or blue-green, because of the possibility that the user is partially or completely color-blind. The red-green should always have some blue tones, and the blue-greens always should have some red tones.

#### **4.3.2.5 Color Selection and Ambient Illumination**

The level of ambient illumination directly affects the perceived brightness and hue of a color. Consider the following when designing a color display:

- Use green, as it provides good general visibility over a broad range of intermediate luminances.
- Use red under high ambient lighting but not in low lighting.
- Use yellow, as it provides good general visibility over a broad range of luminances.

#### **4.3.2.6 Specific Color Meanings**

Use the colors and associated meanings listed below for designing military color coding. The exact color values selected are dependent on the background upon which they are to be displayed.



Color	Meaning
Green	Non-alert, neutral forces, forces or situation at acceptable condition, obstacles on map graphics, ON as opposed to OFF
Blue	Friendly forces symbology, cool, safe, nitrogen, deep
Red	Alert, forces or situation at critical condition, enemy symbology, stop, dangerous, oxygen, hot
Yellow	Forces or situation at marginal condition, unknown forces, caution, NBC areas on map graphics
Black	Political boundary, image or figure edge

#### 4.3.2.7 Using Blue

Blue as a background color is most effective for tasks performed at close distances.

- Because the eye is relatively insensitive to blue, blue lines or dots will be very difficult to resolve. Avoid using saturated blue for small lines or dots when the background is dark.
- Use saturated blue only for background features in a display, not for critical data.

#### 4.3.2.8 Use of Color Keys

While not recommended, there may be some circumstances where the system designer must allow the screen design to deviate from the color meanings provided in the previous lists on classification bar color codes and associated meanings, or use other colors. When this happens, it is important to include on the display a key that explains the color meaning.

- Ensure that the color key is readily accessible visually on the display without having to scroll or expand the screen or window.
- Ensure that the colors in the key have the same appearance as the color being defined.

#### 4.3.2.9 Large-Screen Display Periphery Colors

Avoid the use of red and green in the periphery of a large-scale display. Yellow and blue are good periphery colors.

#### 4.3.2.10 Color Sets

When selecting color sets for displays, ensure that contrast is high between foreground objects and background displays. Black provides high contrast with light shades or with white. No color should be contrasted with a lighter or darker shade of itself, if this can be avoided (e.g., it

cannot be avoided on monochrome displays). The hypertext version of this subsection will include a visual comparison of background versus foreground selections.

### 4.3.3 Tonal Color Coding

#### 4.3.3.1 Color Coding for Relative Values

When relative rather than absolute values of a variable are important, display gradual color changes of a single color as a tonal code to show the relative values of a single variable. Display a monochromatic shading rather than spectral codes (different colors).

#### 4.3.3.2 Ordered Coding

If different map areas are coded by texture patterns or tonal variation, order the assigned code values such that darkest and lightest shades correspond to extreme values of the coded variable.

### 4.3.4 Color-Coded Symbols

Use the following guidelines with symbols that are color-coded.

#### 4.3.4.1 Color-Coded Symbol Size

Ensure that color-coded symbols subtend a minimum of 20 minutes of visual arc. The designer must determine the maximum viewing distance from the display, then calculate the minimum size of the object, using the formula:  $\text{Visual Angle (Min.)} = \frac{(57.3)(60)L}{D}$

where L = size of the object, and D = distance from the eye to the object.

The units of measure can be inches or centimeters (see Figure 4-6).

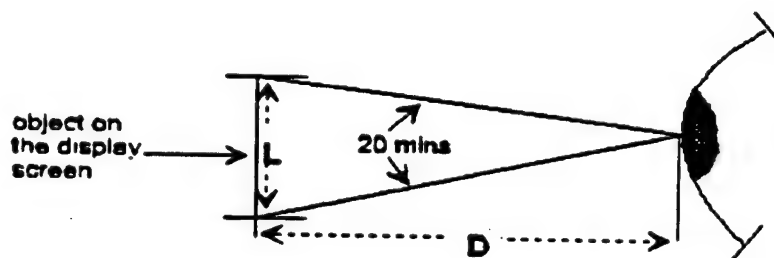


Figure 4-6. Visual Arc Subtended

#### **4.3.4.2 Color-Coded Symbol Brightness**

Ensure that color-coded symbols have a minimum luminance of one footLambert.

#### **4.3.4.3 Refresh Rates**

The minimum refresh rate for color-coded symbols should ensure a flicker-free display. Flicker-free display testing is described in American National Standards Institute (ANSI)/HFS Standard No. 100 (1988).

### **4.3.5 Map Graphics And Color**

#### **4.3.5.1 Functional Versus Decorative Color Coding**

On map graphic displays, use color coding that provides a specific meaning to the user, rather than colors that are decorative only. These specific meanings should be used in accordance with appropriate standards. For example, the U.S. Army standard uses green for vegetation, brown for topographic relief, etc. Standards include U.S. Army FM 21-26, *Map Reading and Land Navigation* (1987); DIA *"DIA Standard Military Graphics Symbols Manual"* (DIAM 65-x) (Draft 1990); and *North Atlantic Treaty Organization (NATO) Standardization Agreement 2019, Military Symbols for Land Based Systems* (1990), available through the U.S. Navy.

#### **4.3.5.2 Differences in Color Perceived Distance**

The designer should be aware of how different colors focus at different distances relative to the user's retina as a result of wavelength. To the user, some colors will appear to be closer than others, especially the more saturated colors.

**This page intentionally left blank.**

## REFERENCES

Paragraph	Reference
4.1	DoD (1992a)
4.2	Galitz (1994)
4.2.1.1	Williams (1987b) Appendix A p. A-1; Smith and Mosier (1986) para 4.0-6; Brown et al. (1983) p. 1-11; Shneiderman (1987) p. 327; Smith and Mosier (1986) para 2.5-1; Lickteig (1989) p.10; Brown et al. (1983) p. 1-1 & 1-11; Tullis (1988) pp. 393 & 336; Hamel and Clark (1986) p. 26; Slominski and Young (1988) p. 2
4.2.1.2	Brown et al. (1983) p. 1-1
4.2.1.3a	Smith and Mosier (1986) para 2.5-2
4.2.1.3b	Hamel and Clark (1986) p. 28; Slominski and Young (1988) p. 3-4
4.2.1.3c	Galitz (1984) p. 103
4.2.1.3d	Galitz (1994) p. 59
4.2.1.4a	Nes (1986) p. 105
4.2.1.4b	Smith and Mosier (1986) para 2.5-10; Lickteig (1989) p. 10; Brown et al. (1983) p.1-5 & 1-12; Shneiderman (1987) p. 327
4.2.1.4c	Bowser (1991) p. 16; Smith and Mosier (1986) para 2.5-11; Galitz (1984) p. 102
4.2.1.4d	Tullis (1988) p. 382
4.2.1.4e	Galitz (1984) p. 102; Lickteig (1989) p. 9
4.2.1.4f	Brown et al. (1983) p. 1-4
4.2.1.5	Smith and Mosier (1986) paras 2.5-16 and 2.5-17
4.2.1.6	Brown et al. (1983) p. 1-11
4.2.1.7	Galitz (1984) p. 102
4.2.1.8	Shneiderman (1992) p. 80
4.2.1.9	Galitz (1994); Shneiderman (1992)
4.2.2	Galitz (1994); Shneiderman (1992)
4.2.2.1	Slominski and Young (1988) p. 2
4.2.2.2	Lickteig (1989) p.9; Brown et al. (1983) p. 1-10; Galitz (1984) p. 99 & 102; Tullis (1988) p. 382

## REFERENCES (cont'd)

Paragraph	References
4.2.2.3	Smith and Mosier (1986) para 2.5-7
4.2.2.4	Galitz (1984) p. 103; Shneiderman (1987) p. 327
4.2.2.5	Smith and Mosier (1986) paras 2.5-13 and 2.5-15; Tullis (1988) p. 387; Shneiderman (1987) p. 336
4.2.2.6	Smith and Mosier (1986) para 2.5-14
4.2.2.9	Shneiderman (1992) p. 84
4.2.3.1a	Bowser (1991) p. 16; Tullis (1988) p. 385
4.2.3.1b	Shneiderman (1987) p. 327
4.2.3.2a	Williams et al. (1987b) Appendix A p. A-1
4.2.3.2b	Smith and Mosier (1986) para 2.5-3
4.2.3.2c	Nes (1986) p. 103; Shneiderman (1984) p. 104; Brown et al. (1983) p. 1-6
4.2.3.2d	Nes (1986) p. 101; Brown et al. (1983) p. 1-13; Shneiderman (1987) p. 105; Tullis (1988) p. 398-399; Grabinger and Amedeo (1988) p. 198
4.2.3.2e	Shneiderman (1987) p. 327
4.2.3.2f	Tullis (1988) p. 395
4.2.3.2g	Galitz (1984) p. 102
4.2.3.2h	Shneiderman (1987) p. 327; Brown (1989) p. 28-29
4.2.3.3a	DoD (1985) p. 3-3
4.2.3.3b	Tullis (1988) p. 399; Galitz (1984) p. 104; DoD (1985) p. 3-3
4.2.3.4a	Shneiderman (1987) p. 104; Nes (1986) p. 112; Tullis (1988) p. 397; Brown et al. (1983) p. 1-9
4.2.3.4b	Galitz (1984) p. 184
4.2.3.4c	Galitz (1984) p. 184
4.2.3.4d	Shneiderman (1987) p. 184
4.2.3.4e	Bowser (1991) p. 16
4.2.3.4f	Bowser (1991) p. 16

## REFERENCES (cont'd)

Paragraph	References
4.2.3.5a	DoD (1985) p. 3-2
4.2.3.5b	DoD (1985) p. 3-3
4.2.3.6	Smith and Mosier (1986) para 2.5-18
4.2.3.7	Smith and Mosier (1986) para 2.5-4
4.2.3.8	Smith and Mosier (1986) para 2.5-5; Galitz (1984) p. 103; Shneiderman (1987) p. 327
4.2.3.9	Smith and Mosier (1986) para 2.5-6; Shneiderman (1987) p. 327; Brown et al. (1983) p. 1-5
4.2.3.10	HFS ANSI 100 (1988)
4.2.3.11	HFS ANSI 100 (1988)
4.2.3.12	Williams (1987b) Appendix A p. A-2; MacGregor and Lee (1988) p. 10; Galitz (1984) p. 103; Shneiderman (1987) p. 336
4.3	Thorell and Smith (1990)
4.3.1	Lickteig (1989) p.10; Nes (1986) paras 4.2.3 and 4.2.4; Galitz (1984) p. 122; Brown et al. (1983) para 7.2; Lickteig (1989) p. 10; Shneiderman (1987) p. 341; Bailey (1982) p. 421; Lewis and Fallesen (1989) p. 20; Rosch (1994)
4.3.1.2a	Smith and Mosier (1986) para 2.6-29
4.3.1.2b	Shneiderman (1987) p. 72 and 337; Tullis (1988) p. 390
4.3.1.2c	Lewis and Fallesen (1989) p. 23; Nes (1986) para 4.1.1; DoD (1989) para 9-1.3.3; Galitz (1984) p. 126 and 127; Slominski p. 4; Matthews (1987); Sidorsky p. 6.3-15q
4.3.1.2d	Galitz (1994), pp. 377-402; Galitz (1984) p. 126-127
4.3.1.2d1	Sidorsky p. 2.3.6 q-6,7; Galitz (1984) p. 121; Bailey (1982) p. 43; Lewis and Fallesen (1989) p.25
4.3.1.2d2	Smith and Mosier (1986) para 2.6-30; Brown para 7.4; DoD (1989a) para 5.4.1.4.5.5; Bailey (1982) p. 63; Lewis and Fallesen (1989) p. 20; Lickteig (1989) p. 10
4.3.1.2d3	Brown et al.(1983) para 7.7.6
4.3.1.2d4	Bowser (1991) p.18; Lewis and Fallesen (1989) p. 21; HFS (1988)

## REFERENCES (cont'd)

<b>Paragraph</b>	<b>Reference</b>
4.3.1.2d6	Bailey (1993)
4.3.2.1a	Smith and Mosier (1986) para 2.6-27
4.3.2.1b	Galitz (1984) p. 125; Lewis and Fallesen (1989) p. 22
4.3.2.1c	Galitz (1984) p. 123
4.3.2.1d	Lewis p. 22
4.3.2.1e	Smith and Mosier (1986) para 2.6-26, 31; Sidorsky p. 6.3-15 o-1; Brown para 7.6.1; Nes (1986) para 4.2.2; Galitz (1984) p. 122; DoD (1989) para 5.15.3.3.7; Lickteig (1989) p. 10; Shneiderman (1987) p. 339
4.3.2.2a	Bowser (1991) p. 18; Brown (1983) para 7.6.2; Galitz (1984) p. 125; Shneiderman (1987) p. 340; Lewis and Fallesen (1989) p. 21; Bailey (1982) p. 263
4.3.2.2b	Bailey (1982) p. 246
4.3.2.2c	Smith and Mosier (1986) para 2.6-32; Sidorsky p. 6.3-15 o-2 and 2.3.6 q-3, 4; Brown (1983) para 7.7.1; Galitz (1984) p. 125; DoD (1989) para 5.2.2.1.18; Shneiderman (1987) p. 340; Hamel p. 5; Bailey (1982) p. 246; Lickteig (1989) p. 10; U.S. Department of the Army (1985b)
4.3.2.3a	Smith and Mosier (1986) para 2.6-28; Brown (1989) para 7.1; Galitz (1984) p. 127; Lickteig (1989) p. 10; Smith and Mosier (1986) para 2.6-28; Chao (1987) p. 361; Lewis and Fallesen (1989) p. 20
4.3.2.3b	Galitz (1984) p. 127; Nes (1986) para 4.2.2 and 4.2.5; Slominski p. 4; Shneiderman (1987) p. 338
4.3.2.3c	Sidorsky p. 6.3-15 o-3; Galitz (1984) p. 125; Lickteig (1989) p. 10; Bailey (1982) p. 421; Shneiderman (1987) p. 71
4.3.2.4a	Lewis and Fallesen (1989) p. 22
4.3.2.4b	Galitz (1984) p. 126
4.3.2.4c	Galitz (1984) p. 126
4.3.2.4d	Shneiderman (1987) p. 341; Snyder (1988) p. 465; Matthews (1987) p. 23



## REFERENCES (cont'd)

Paragraph	Reference
4.3.2.4e	IBM (1984) p. 19
4.3.2.5a	Galitz (1984) p. 127
4.3.2.5b	Galitz (1984) p. 127
4.3.2.5c	Galitz (1984) p. 127
4.3.2.6	Sidorsky p. 2.3.6 q-3 and 4; DoD (1989) p. 256; U.S. Department of the Army (1985) p. 2.2
4.3.2.7	Lewis and Fallesen (1989) p. 23
4.3.2.7a	Snyder (1988) p. 465
4.3.2.7b	Smith and Mosier (1986) para 2.6-34; Brown (1983) para 7.7.5; Galitz (1984) p. 127
4.3.2.8	Sidorsky p. 6.3-15 o-4; Brown para (1983) 7.6.1; Nes (1986) para 4.2.4; Galitz (1984) p. 123; Lickteig (1989) p. 10; U.S. Department of the Army (1985) p. 2-2
4.3.2.9	Lewis and Fallesen (1989) p. 22
4.3.2.10	Lewis and Fallesen (1989) p. 22; Shneiderman (1987) p. 341; Snyder (1988) p. 465; Sidorsky para 2.3.6; Thorrell p. 2 and 3
4.3.3.1	Smith and Mosier (1986) para 2.6-25
4.3.3.2	Smith and Mosier (1986) para 2.4.8- 7
4.3.4.1	Durrett (1987) p. 186; Van Cott and Kinkade (1984) p. 47
4.3.4.2	Breen et al. (1987) p. 207
4.3.4.3	Breen et al. (1987) p. 209; HFS ANSI 100 (1988)
4.3.5.1	Olson (1987) p. 207; U.S. Department of the Army (1987); DIA 1990; NATO 1990
4.3.5.2 Olson (1987)	p. 20

**This page intentionally left blank.**

## 5.0 WINDOWS

A window provides the visual means by which the user can interact with an application program. A window displays the results of the command or data input by keyboard, mouse, or other device. A window display screen is analogous to a window in a wall that allows one to see into a room; the window display screen allows the user to see into a software program. A window is typically rectangular and can cover part or all of a display screen. In addition, multiple windows on a display can be open at one time. Figure 5-1 illustrates an OSF/Motif window. Windows for other GUI, such as OS/2 and Windows, have similar though not identical characteristics. Arrows and labels identify key parts of the window. Section 5.0 provides general guidelines for windows. Refer to commercial GUI style guides for specific window design details and explanations of attributes and terms used to describe the actions, warnings, and information presented to the user.

Commercial GUI designs provide a number of basic functions, allowing the user to control window operations. While each GUI provides its own specific functions, the following are typical examples. By opening and closing a window, a task or application is started, stopped, or removed from the screen. Scrolling allows the user to view the information within a window, including that which is outside the normal boundaries of the window. Windows can be stacked on top of each other like paper on a desk. Good designs should provide the user the capability to access available GUI functions when they are required.

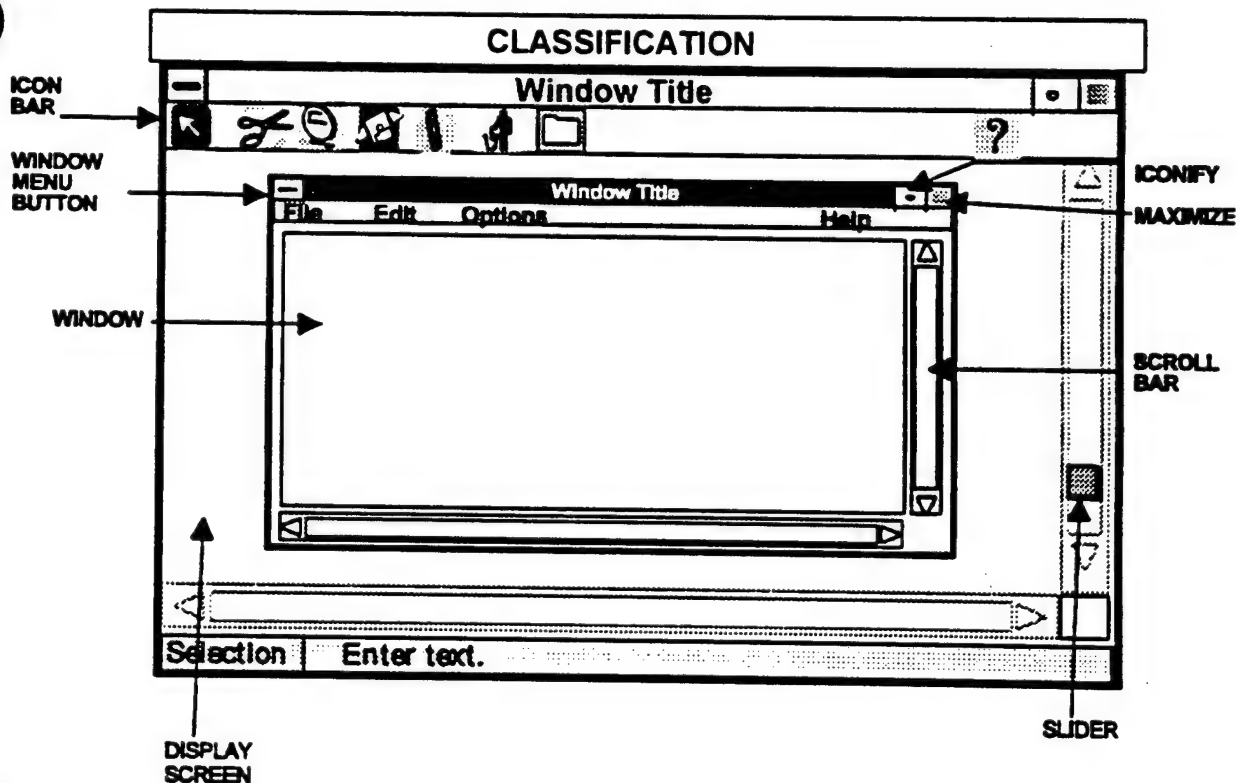
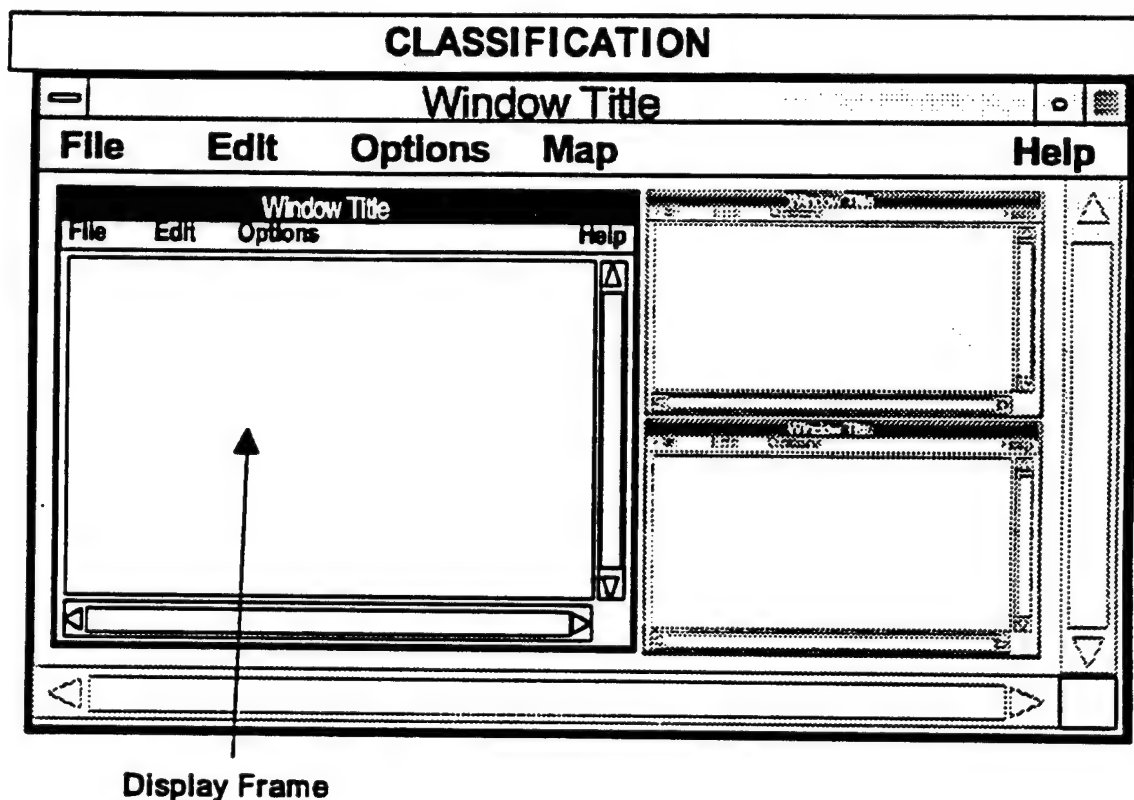


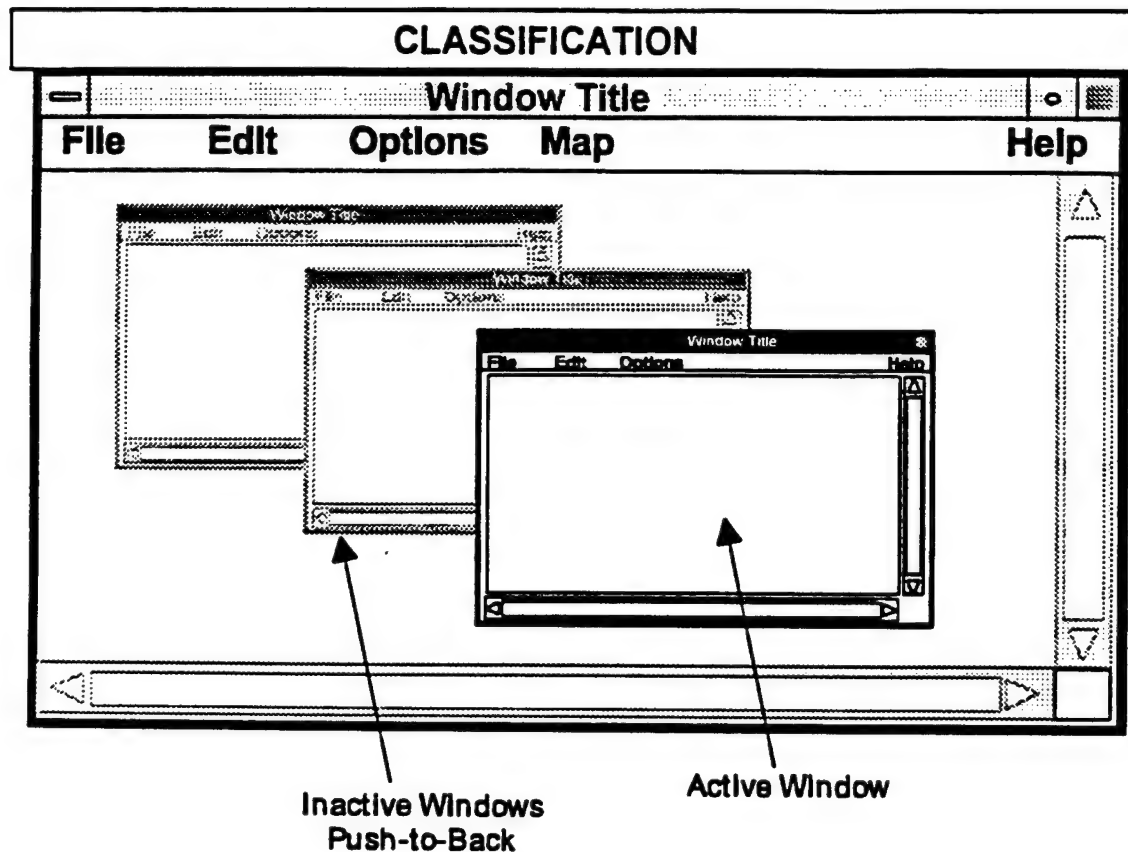
Figure 5-1. Example of a Typical Windowing Screen

The two basic approaches to simultaneous window presentation are tiling and overlapping. In the tiling approach, multiple windows do not overlap but lie on the same plane. Their borders are flush, and developers usually limit the primary control operations to designation and scrolling, while limiting or blocking basic functions, such as opening and closing or moving and sizing. Figure 5-2 illustrates the tiling type of window design.

Using the overlapping method, windows are presented on multiple planes and appear to be 3-dimensional. Windows can overlap or even obscure each other, like pieces of paper on a desk top. The window made active by the user will appear on top, pushing inactive windows to the back. The user normally has access to all previously discussed control functions to control overlapping windows. Figure 5-3 illustrates this type of window design.



**Figure 5-2. Example of the Tiling Approach**



**Figure 5-3. Example of the Overlapping Approach**

The design of application windowing interfaces should begin with the screen design principles given in Section 4.0 of the *Style Guide* and should address the basic window attributes using the appropriate commercial style guide. The depth and breadth of research on the impact of windows on user performance are not as great as they are for other design areas (Billingsley 1988).

In general, the following generic guidelines should be applied to window GUIs.

- Be consistent in how the windows look and “act.”
- Recognize the limitations that the specific system hardware imposes on the usefulness of windowing software. For example, ensure that the display device has the resolution and size to properly support a windows approach to information presentation. When the hardware will not adequately support the windowing environment, alternate hardware should be considered.

- Ensure that the central processor unit (CPU) has the power, in terms of memory and speed, to effectively use a windows approach. Without a proper CPU, slow system-response time will significantly degrade the speed of information presentation. Again, when the hardware will not adequately support the windowing environment, alternate hardware should be considered.
- The windows interface is especially important when the user needs to perform multiple tasks or see different sets of data concurrently.
- Each open window requires system resources in terms of memory and processing speed. Through experimentation, determine a limit on the maximum number of windows that can be effectively opened for each system.

## 5.1 WINDOW BASICS

A clarification of window and screen-related terms is provided in Figure 5-4. The Glossary, Appendix B, contains additional term definitions. The *Style Guide* should not conflict with commercial GUI style guides, and the actual definition to be used by the developer should comply with the selected commercial GUI.

### 5.1.1 Basic Window Appearance

The displayed window appearance is determined by the selected GUI and related commercial style guide. Specific domains further define window appearance, such as intelligence, where the basic CMW window components are added to the commercial window design. The classification bar displayed as the top line of the basic window and the optional input information label displayed at the bottom of the screen are additional features supported by the intelligence community CMW operating system rather than by the GUI style selected. However, from the CMW application designer's viewpoint, classification bar and input information label are displayed in the same manner as other window controls (e.g., the title bar). Appendix A includes a detailed description of the fields that make up these security bars. Until the CMW becomes an integral component of the DoD system architecture, each DoD organization should adhere to its own security standards.

#### 5.1.1.1 Title Bar

The appearance of the title bar and associated controls are determined by the selected GUI. The creation of the titles within a title bar is subject to general positive design principles. If the application contains multiple primary windows (e.g., to display different files), then the window title should include the application name and the name of the currently displayed file.

Category	Term	Definition	Reference
Screen-related terms	computer screen	Hardware monitor total display area	Figure 8.5
	display frame	Area of a screen identified for design, that includes more than one window or object	Figure 5.2
	display screen	Area of the hardware screen used for display	Figure 5.1
	dialog box	Screen display box containing a message requesting additional information from the user	Figure 5.7
	input focus	Applies to window that actually receives user input. Input focus may be explicit or implicit (see glossary).	Figure 5.11
	window	Typically rectangular display that provides a visual means for interaction with an application	Figure 5.1
Window-related terms	multiwindow	Simultaneous display of several windows on the computer screen	Figure 5.2
	push-to-back	Process of moving a window to the background	Figure 5.3
	overlapping	Windowing system in which one window covers a portion of another	Figure 5.3
	tiling	Windowing approach in which multiple windows do not overlap, rather, all lie on the same plane	Figure 5.2
Window/screen parts	cursor	Visual mechanism to mark, on-screen, where current input or output is to happen	Figure 9.3
	pointer	Graphic on the screen display that represents the mouse or trackball position	Figure 5.11
	resize border	Window border that, if selected, allows user to resize the window	Figure 5.14
	scroll bar	Rectangular bar that may be along the right edge or bottom of a window. Clicking or dragging in the scroll bar causes the view of the document to change.	Figure 5.1
	slider	Part of the scroll bar that indicates what part of the file contained in a window is being viewed	Figure 5.1
Menu-related terms	menu	List of options available within a software application	Figure 6.5
	icon bar	Horizontal or vertical layout of icons used as buttons to quickly access frequently used commands and macros	Figure 5.1
	menu bar	Horizontal menu, usually at the top of the screen, that contains menu titles	Figure 5.8
	menu button	A button in a standardized location used for window management functions (i.e., close, move, resize)	Figure 5.1
	hierarchical menu	Method of organizing menus in layers. The secondary or tertiary menus are stored within a primary menu.	Figure 6.7
	pop-up menu	Lists of options that appear on the display screen in the form of a window	Figure 6.3
	pull-down menu	Lists of options attached to a selection on a menu bar	Figure 6.2

**Figure 5-4. Window and Screen-Related Terms**

Some general considerations that apply to creating titles are the following:

- Include the name of the application in the title, followed by a colon, followed by the name of the currently displayed file (e.g., Editor: Myfile.txt)
- Center the title
- Distinguish the title by a visual attribute (e.g., boldface type)
- Use application and function name to identify an open window, not system-level window name (e.g., messages:e-mail as opposed to ATCCS:e-mail)
- If the selected commercial GUI allows, enable the window title to display the version number of the application, but do not use the window title area to display any messages.

#### **5.1.1.2 The Window Menu Button**

The window menu button for Windows and Motif systems is located in the upper left-hand corner of the title bar (see Figure 5-1). This button provides a standard location for window management functions (e.g., close, move, and window resizing functions). A more detailed explanation of the functions and features supported by the window menu button can be found in the relevant GUI style guides. The principle of using a consistent location and shape for standard controls should be applied to all applications.

#### **5.1.1.3 Reducing the Window to an Icon**

In standard GUI styles, users can iconify windows. For example, if the user is not actively using a base window but wishes to maintain easy access to it, or if the window is active but does not require user interaction for extended periods, these windows can be iconified. If a window is reduced to an icon, the window is removed from the screen, and the application controlling the window is represented as an icon. Application processing can then continue in the background, as if the window were still displayed on the screen. This capability to iconify is available to application developers as part of any standard GUI implementation and should be used as appropriate to good user interface design.

#### **5.1.1.4 Expanding a Window to its Full Size**

Expanding a window to its full size (maximizing) increases the size of the window to the maximum specified by the application. The maximize methods used by the various GUIs include selecting a maximize button, selecting a maximize function from the window menu button, or depressing the maximize accelerator keys with the window focus appropriately selected. Windows can also be expanded to full size by dragging (see Paragraph 5.1.2) the resize borders or resize corners. The capability to easily expand a window to full size is



available to application developers as part of any standard GUI implementation and should be used when appropriate to good user interface design.

### **5.1.2 Dragging the Window**

Dragging refers to a user's ability to reposition windows or window borders on the screen. Dragging a window moves it to a different position on the computer screen. As the window is dragged (or moved), a "ghost" outline of the window should move with the pointer. The window should move to the position of the outline when the procedure is complete (e.g., mouse button is released).

### **5.1.3 Scroll Bars**

The scroll bar is a special type of control that makes it easy for the user to view or page through objects such as documents, drawings, and spreadsheets too long or wide to be displayed in the application area, also called a pane. Scroll bars also aid users in panning graphic map displays in the north/south and east/west directions. Scroll bars give users the capability to navigate through documents without paging one window at a time. This interface capability is available to application developers as part of any standard GUI implementation and should be used when appropriate to good user interface design for all windowing applications.

### **5.1.4 Application Area**

The application area or pane is the part of the window where applications display and collect data and where users perform most application tasks. For example, if a user is working with a text editor, the application area could contain the document to be edited. When using a windowing interface, the application area should be clearly and consistently identified to the user.

### **5.1.5 Message Area**

The message area (or footer) is reserved for noncritical application messages that should not suspend processing. Use the left side of the message area for short-term messages, such as "Incorrect format - field requires numeric data. Please reenter." Use the right side of the message area for medium-term messages, such as "Page 4 of 29."

### **5.1.6 Resizing The Window**

The application suggests the initial size of its window to the window manager. Because work and preferences vary, users should generally be able to alter the size of windows. Resizing windows is performed through "hooking" the edge or corner and dragging the cursor to reduce or increase the window size, or by using buttons typically located in the upper right corner of a window. Resizing a window normally increases or decreases the size of the window frame, not the scale of the data within the window. For example, if a window containing a text document is enlarged, more lines of data may be seen, but the text itself does not enlarge.

The specific resizing behavior of a window is determined by the commercial GUI selected. However, the following are general principles to use when resizing windows:

- In the minimum height of a window, allow enough room for at least the classification bar, title bar, and menu bar (control area).
- Design the application logically to accommodate the resizing function. Include important information in the upper left-hand corner of the window.
- When a user resizes a window, ensure that only the size of the window's borders changes, not the size of graphics, text font size, relative position of the data, or the controls within the borders. The normal result will be an increase in the amount of viewable text or number of objects in the window. An exception might occur in imagery manipulation where the user may require the image to rescale (magnify) with the window frame.
- Ensure that resizable windows are easily distinguishable from those that cannot be resized, such as the system window.

#### **5.1.7 Window Controls**

Controls and their labels represent application functions in windows and dialog boxes. See Subsection 6.6 on Dialog Boxes/Pop-Up Windows.

- Controls should mimic the physical items they represent (e.g., switches or buttons) by providing feedback before, during, and after their selection by a user. For example, a button that the user has chosen should appear to be pushed in.
- Window controls are usually activated using the SELECT button on the pointing device. However, users who interact with the application using only the keyboard should have equivalent functionality. Ensure that control appropriate to the selected commercial GUI is available. Control examples include the "TAB" or arrow keys, allowing the user to move between controls and using the Return/Enter key to invoke the default of the indicated control. Also, when mnemonics are available to application developers as part of a standard GUI implementation, the keyboard user should be provided with mnemonics for each control.
- Graphic display of a control should use shape, shading, outline, and (when appropriate) color to aid the user in identifying the active control area. When the control background is the same as the area surrounding the control, care must be taken to clearly mark the control area for easy identification.

#### **5.1.8 Window Colors/Patterns/Audio Signals**

The proper use of color, background patterns, and sound may significantly aid the user. This section provides recommendations for using these features.

- Ensure that color is always redundant with some other visual attribute; color should not be provided as the only means of visual distinction.
- On both color and monochrome displays, use background patterns to highlight, group, or clarify relationships and to add extra meaning.
- For quick and accurate interpretation, use colors sparingly and ensure that these colors match user expectations (see Subsection 4.3).
- Ensure that colors that may be changed are not “hard coded” into applications. When appropriate, users should have the option to select their own color schemes (see Subsections 4.3 and 14.1).
- Some colors have strongly associated meanings that the designer must be aware of and use, not abuse. For example, a user may assume that a red control button has critical or irreversible consequences. Thus, avoid red for noncritical buttons, as it may inhibit the user from exploring them. Some common color meanings are as follows:
  - Red            Stop, alarms, errors, danger, critical consequences
  - Yellow        Warning, caution, approaching critical
  - Green        Normal, safe, within normal range, proceed
  - Blue         Cold, water, noncritical items
  - Gray         Inactive, unavailable options or actions.
- Use both color and sound for messages that require user acknowledgment. Display critical messages using red (i.e., borders, text background) and continue the audio alarm until the user responds. Display noncritical messages (e.g., “Printer error. Please check printer and retry or cancel”) using yellow (i.e., text background, graphic, border, etc.) accompanied by a short audio alert.
- Do not use spectral extremes (e.g., red and green, see Subsection 4.3) on a display at the same time, close together. Colors at considerably different wavelengths appear to vibrate when placed together.
- When data are color-coded, provide a legend (e.g., “Orange = Required Field”) at the bottom of the window. Limit color codes to four per window and no more than seven per application.
- Use the same color scheme (i.e., window background, foreground, etc.) for all windows of an application. Repeated use of the same color for similar user interface components or data types allows quick association of elements.

- White text on a black background produces halation, or the spreading of light, making the text less readable. Displaying text in multiple colors also makes text less readable and should be used only if the addition of colors provides significant additional meaning.
- Ensure that the computer screen and window pane (workspace) background is appropriate to the expected lighting conditions (see Subsection 4.3) and that it is a neutral color.
- Ensure that the application window borders contrast sufficiently to stand out from the screen background. At the same time, provide a neutral back-ground for the application data to ensure readability. Muted pastels are recommended.
- In general, the larger the object, the less saturated or deep its color should be to avoid eye fatigue.
- CMW Classification Bar colors are listed below. Environments that use DoD security classification colors should restrict background colors that match the domain-level definition of these display colors.
  - Green      Unclassified
  - Blue      Confidential
  - Red      Secret
  - Orange      Top Secret
  - Yellow      Sensitive Compartmented Information.

## **5.2 WINDOW DESIGN**

### **5.2.1 General Guidance**

#### **5.2.1.1 Hardware Limitations on the Use of Windowing**

When the interface is affected by limitations of the selected hardware platform(s), the developer needs to design the windowing interface accordingly. Hardware limitations to consider include:

- Small screen size, resulting in frequent manipulation of the screen by the user
- Slow processing speed, resulting in slow operation of real-time applications performed by the computer
- Low screen resolution, resulting in less effective visual coding, especially for graphical interface presentations such as symbols and icons.

### **5.2.1.2 Flexibility of Window Specification**

A key to the effective design of a windowing user-computer interface is the flexibility the user has in customizing window content and format. A balance must be achieved between user-specified windows and preformatted windows.

- When the need to view several different types of data jointly cannot be determined in advance, allow a user to specify and select separate data windows that will share a single display frame.
- Where the information required for decision-making may vary according to the situation, allow the user to specify what information to include in a display.
- When content of particular operational displays can be determined during interface design, provide the user with preformatted windows, such as standard message texts for data entry and display.
- Allow the user to display several of these windows concurrently, according to the operational need.

### **5.2.1.3 Temporary Window Objects**

Temporary window objects (e.g., pop-up menus or data, option menus, data filters) are especially effective for providing a menu of alternatives for field entry in preformatted tactical messages and database queries.

- When it is necessary to temporarily add requested data or other features to a current display, provide window objects for that purpose.
- Ensure that a temporary window object does not completely cover the active window, thereby obscuring critical control information and command entry widgets, soft keys, or other activation points.
- When a window object temporarily obscures other displayed data, ensure that obscured data are not permanently erased but will reappear when the object is later removed.

### **5.2.1.4 Number of Allowable Open Windows**

To ensure that system response time is not compromised, design into the interface a defined upper limit on the number of windows allowed to be open at one time.

### **5.2.1.5 Window Physical Design**

- Avoid visual clutter in designing windowing systems.

- For tiled window systems, minimize the clutter at the edges caused by scroll bars, etc. Figure 5-5 illustrates a cluttered window design for tiled windows. Figure 5-2 illustrated an uncluttered display.
- For overlapping window systems with multiple windows, keep back-ground pattern neutral, rather than use complex patterns. Figure 5-6 illustrates a cluttered display; Figure 5-3 illustrated an uncluttered display.
- When a display window must be used for scanning data exceeding more than one line, ensure that the window can display more than one line of data.
- When the system provides an area within a window for command entry, messages, or prompts, place this area as specified in the commercial style guide, or if not specified in commercial style guide, place this area at the bottom of the window display.
- Dialog boxes should be designed to comply with the selected commercial GUI to ensure that they look and function consistently for all applications and systems. To achieve this, follow these recommendations. See example in Figure 5-7.
  - Control buttons used to input a command from a dialog box should be located consistently, for example at the bottom of the window. If the selected commercial GUI allows this placement, this is consistent with the user's natural task flow.
  - The button used to input the selected or default command (usually an OK) should be located consistently, normally on the left side of the box. The CANCEL button is usually located on the right side. Any additional control buttons should generally be located between the OK and CANCEL buttons.
  - The individual commercial GUI style guide is the primary source for the specific placement of controls in a dialog box.

## **5.2.2 Window Control**

Control refers to how the user manipulates the window, not how the application operates within the window. Guidelines for the design of window control fall into five basic topics: general guidelines, opening and closing, moving, sizing, and scrolling.

### **5.2.2.1 General**

- When a user may perform application control actions (such as command entry) while working within a window, ensure that those control actions will be consistent from one window to another.
- Ensure the means provided to the user for controlling (after initial display) the size, location, and characteristics of superimposed window objects operate consistently from one display to another for each type of object.

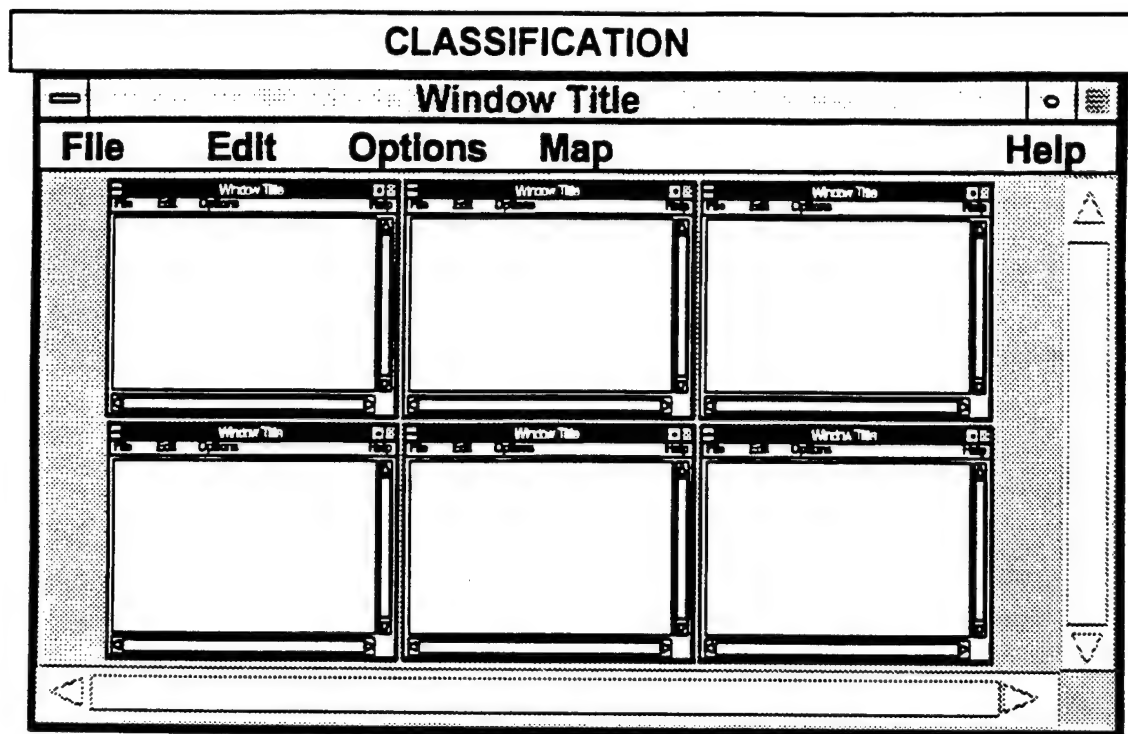


Figure 5-5. Example of Cluttered Window Design for Tiled Windows

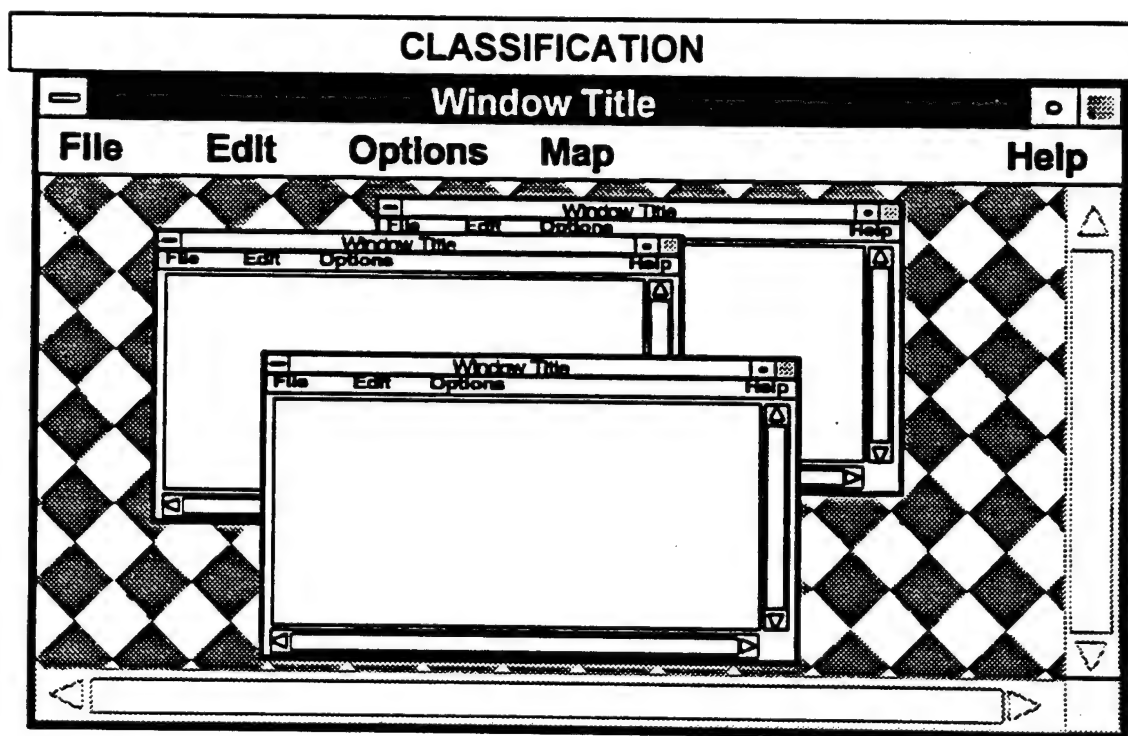
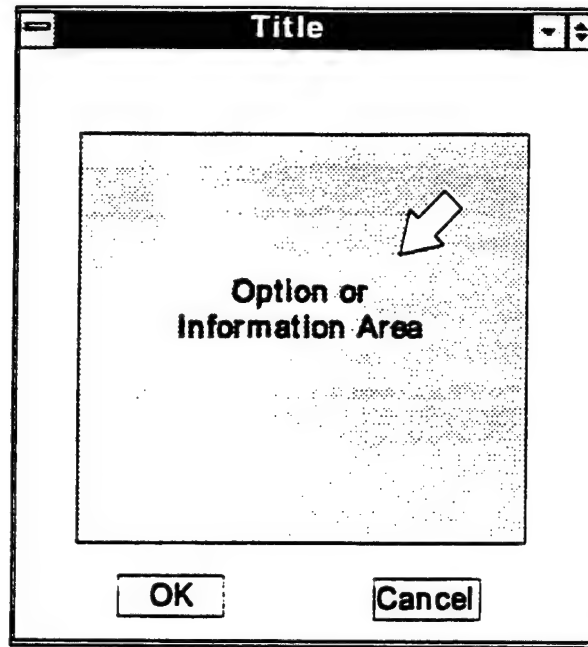


Figure 5-6. Example of Cluttered Window Design for Overlapping Windows Induced by a Complex Background Pattern



**Figure 5-7. Example of a Dialog Box Design**

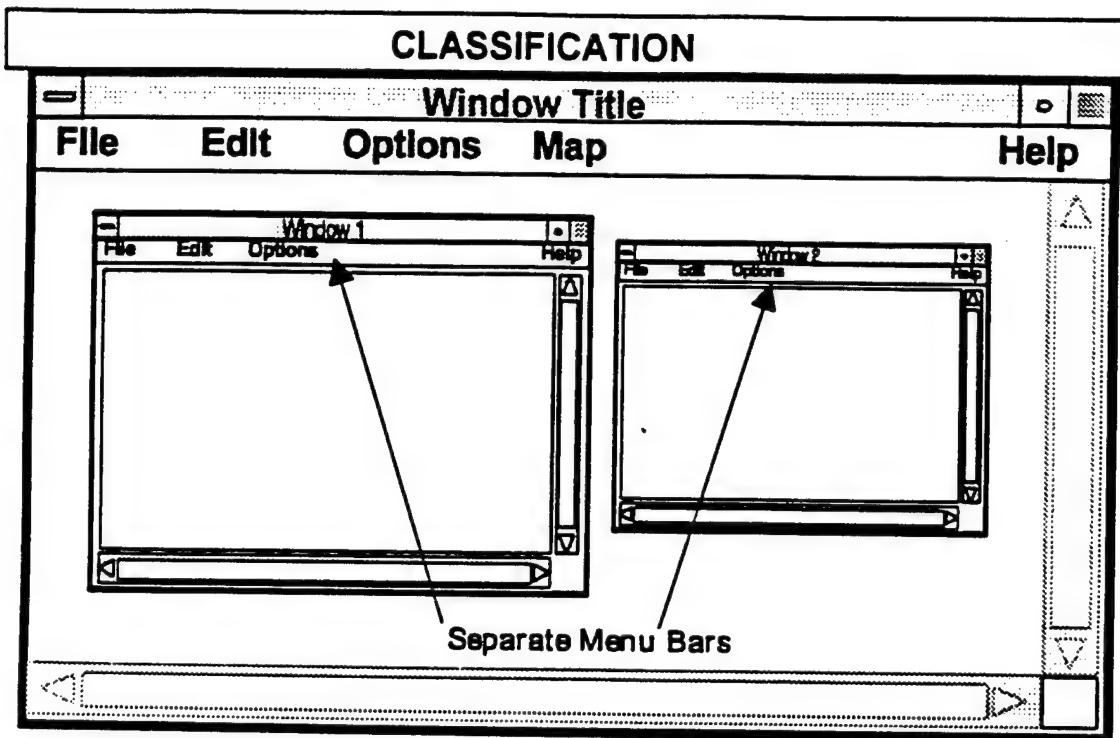
- Provide an easy means, such as iconization or closing, for the user to suppress the display of window objects.
- Provide a separate menu bar for each application window, where different applications are operating concurrently in open windows (e.g., multitasking). See the example in Figure 5-8.

#### **5.2.2.2 Opening and Closing Windows**

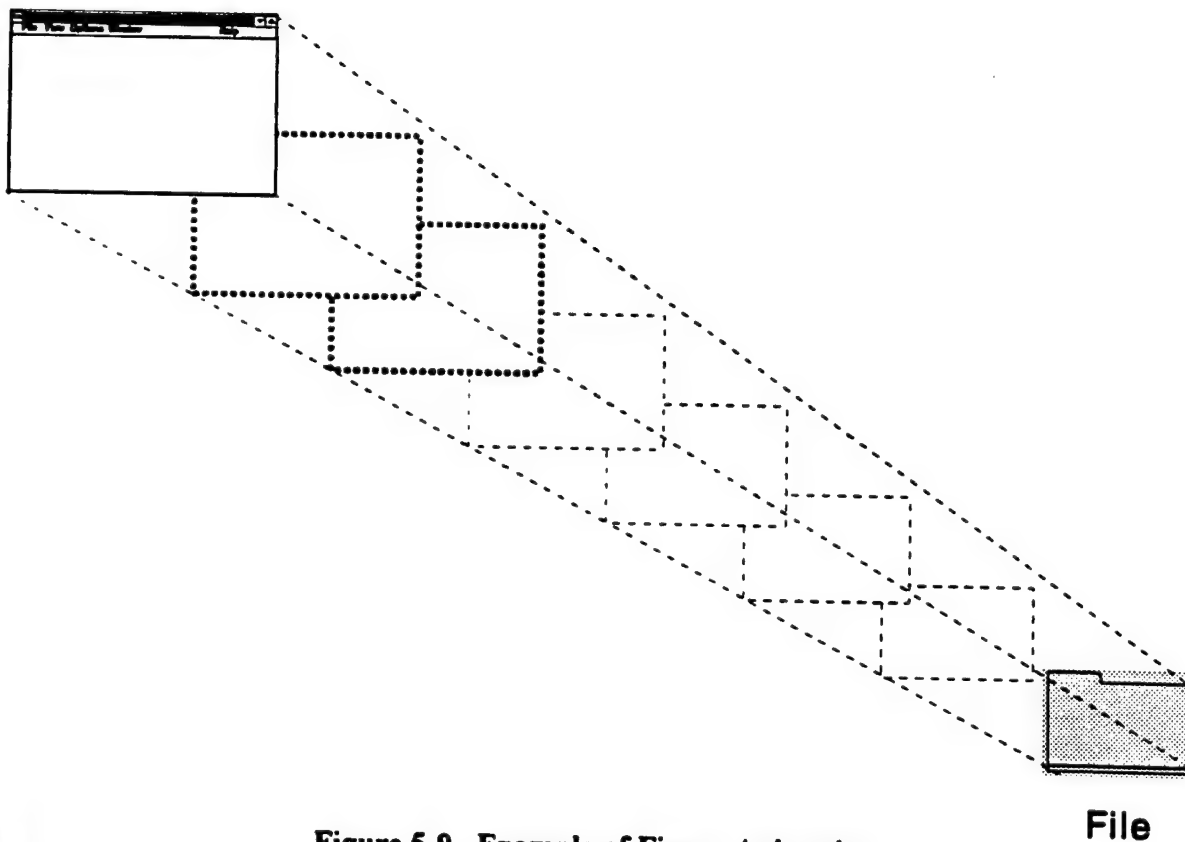
Windows can be opened or closed by menu selections, or a close-button widget (i.e., a small, push-button control object usually located in the upper left corner of a window), or opened from an icon or minimized to an icon. When designing the opening and closing operations, consider the following guidelines:

- The software should provide an animated depiction of opening and closing a window by portraying the window shrinking to an icon and vice versa. This helps the user relate the window, icon, and action (see Figure 5-9).
- When a main applications window is closed by the user, all associated subordinate windows and dialog boxes should also close.





**Figure 5-8. Example of Different Applications with Separate Menu Bars**



**Figure 5-9. Example of Figure Animation**

### 5.2.2.3 Moving Windows

- Provide either full movement of the window (see Figure 5-10) or move an outline, leaving the window visible on the screen.

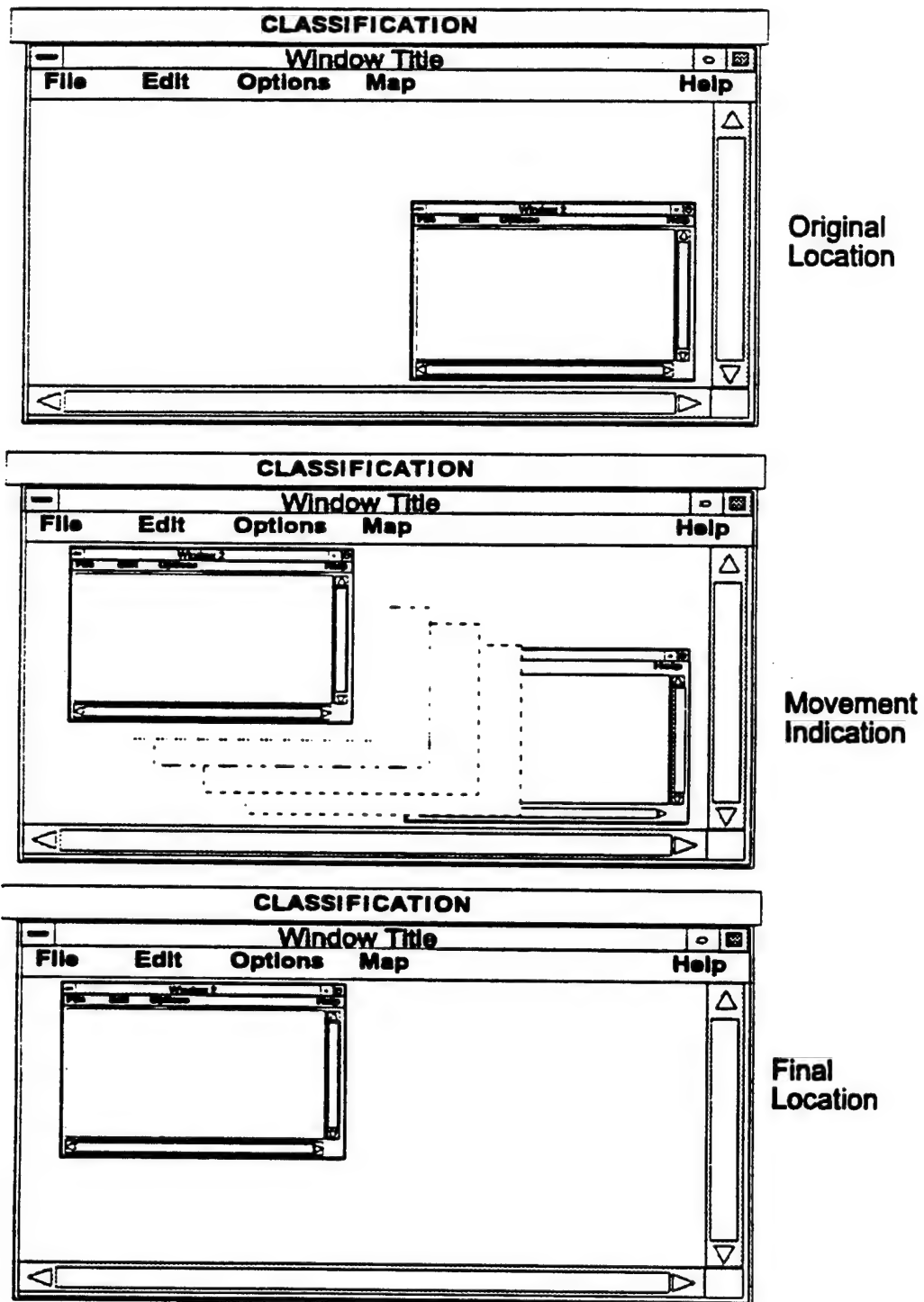


Figure 5-10. Example of a Screen Move

- When the user must select a specific move function to relocate a window on a screen, ensure that the cursor indicates this by a change in shape. Figure 5-11 illustrates one type of cursor change.

#### 5.2.2.4 Resizing Windows

- Provide system protection against obscuring critical control information during window manipulation, especially during user maximization of the window. This means system protection for both the data being retrieved through dialog boxes and system-level control information, such as alert indications. See Figures 5-12 and 5-13.
- When a window is resized, ensure the window contents remain visible during the resizing to provide a visual indication of the effect on the window contents (e.g., visibility and integrity of the image), rather than providing just an outline. Keeping the contents visible will reduce the number of steps required by the user (e.g., resize, view, etc.).
- Resizing of tiled windows by a user is not recommended. If resizing is absolutely required for a tiled window system, ensure that the system automatically resizes all other open windows when one is resized by the user.

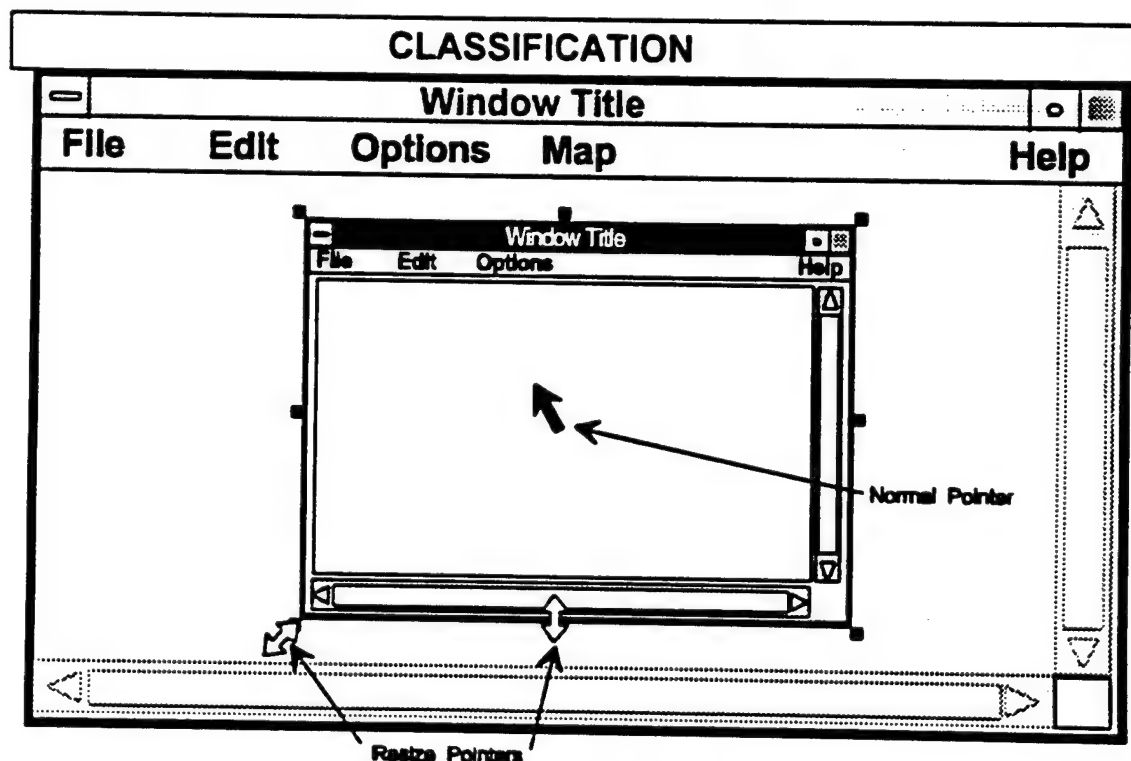


Figure 5-11. Example of a Pointer Changing Shape

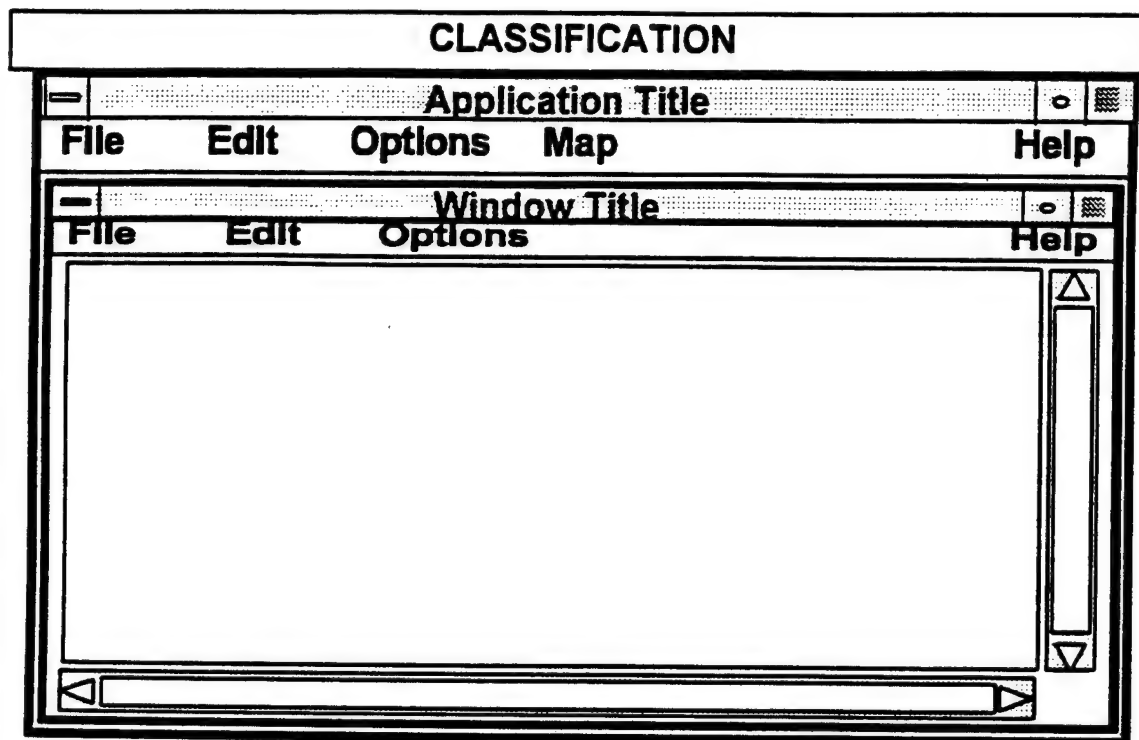


Figure 5-12. Maximum Size of Window

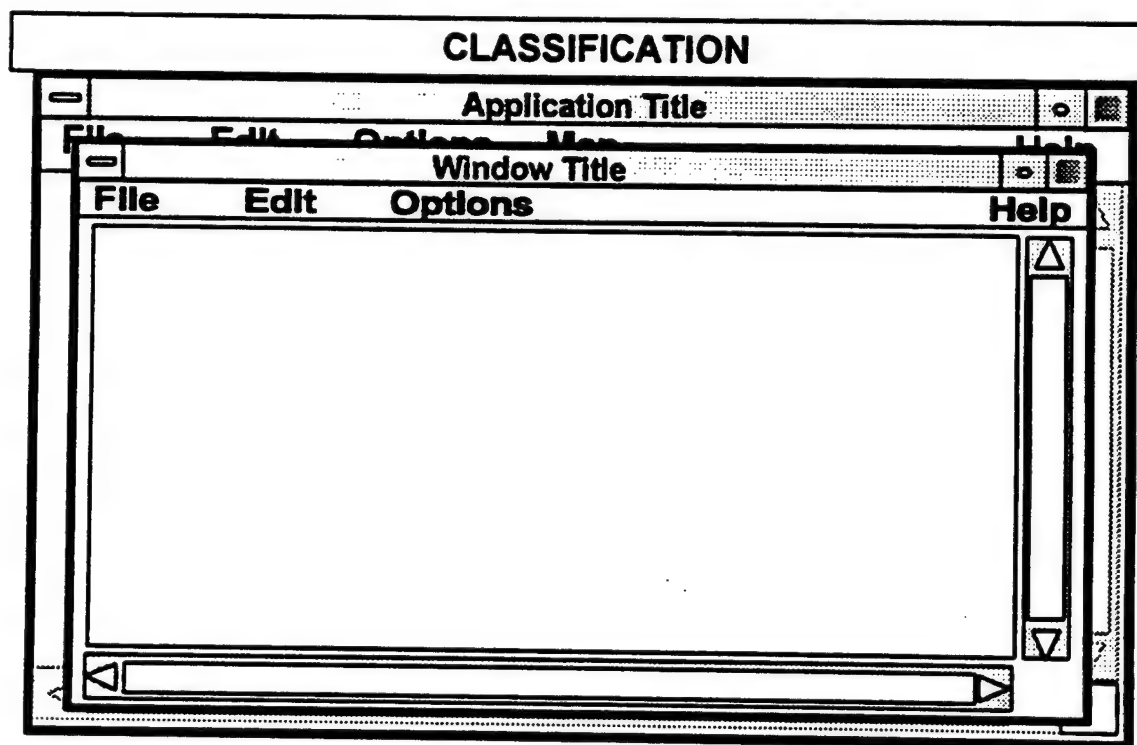
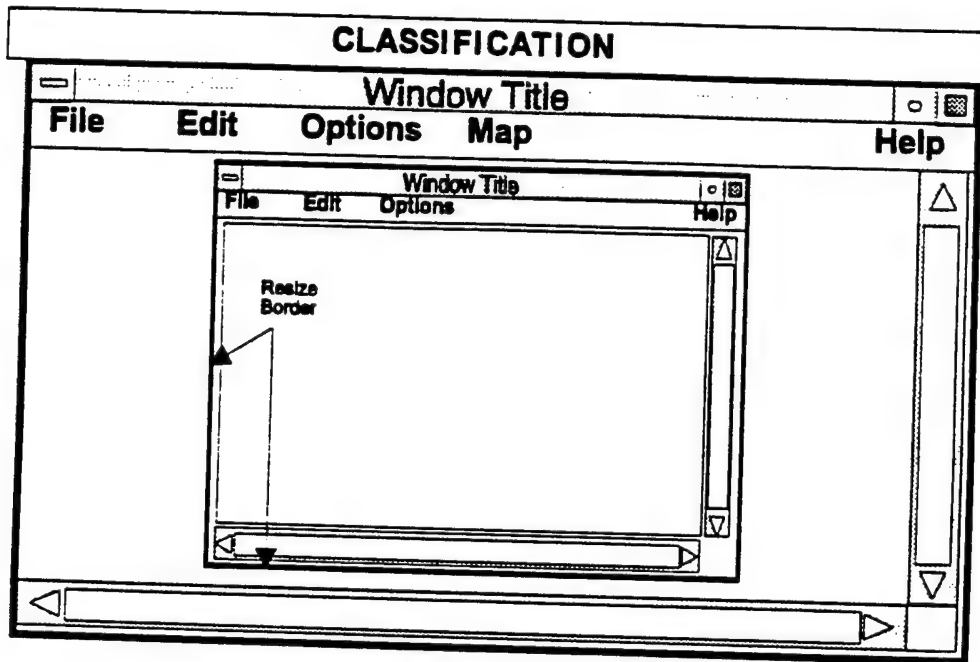
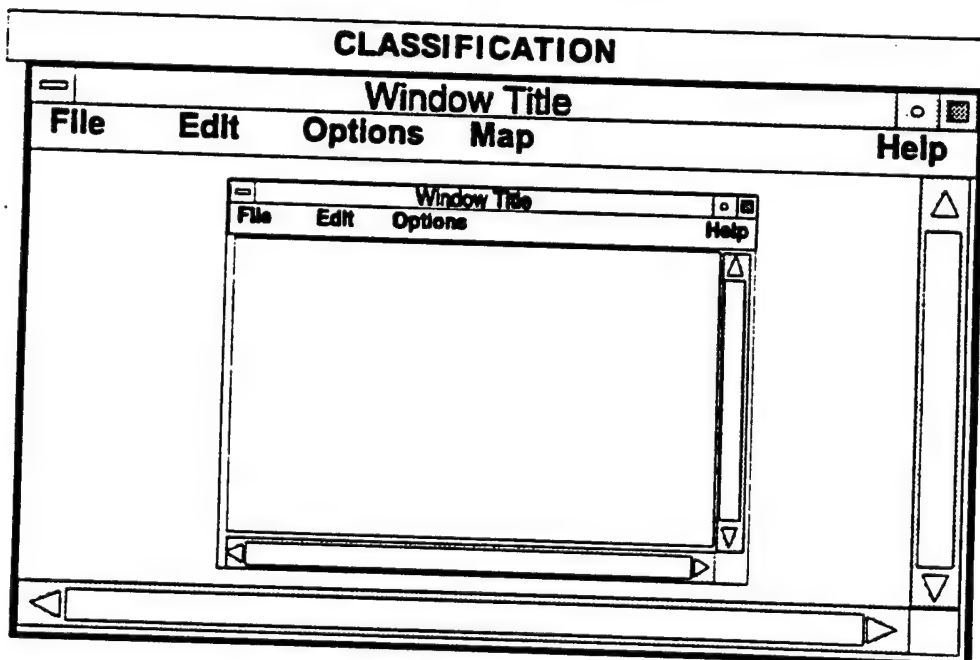


Figure 5-13. Window Size Too Large, Covering Critical Information

- Most windows have a resize border (see Figure 5-14) located at the peripheral edge. If a window cannot be resized, the resize border should be removed to provide a positive indication to the user that the window size is static.



Window with a Resize Border



Window with Resize Border Removed

Figure 5-14. Resize Border Removal

### 5.2.2.5 Scrolling Windows

Scrolling a window can be performed two ways: 1) move the window over the data, where upward movement of the scroll bar causes data to appear to move down; 2) move the data past the window, where upward movement of the scroll bar causes data to appear to move up.

- For scrolling, the system should move the window over the data, as this is consistent with the general convention in industry, such as found in the *OSF/Motif Style Guide*.
- The distance the slider moves on a scroll bar should be proportional to the distance traveled through the file in a window to assist the user in determining current location relative to the total file.
- Design window displays to preclude excessive scrolling. If possible, use a single screen for the full display, unless it causes reading difficulty due to reduction of screen character size.
- Do not display the scroll bar if scrolling is not necessary.

### 5.2.3 Designation

Designation is the process of selecting and indicating with visual cues which window the user can use. This window is called the input focus window.

#### 5.2.3.1 Positive Indication of the Active Window

When more than one window is open, provide the user with a clear, positive indication of the active window by means of a more complex border, subtle change in color hue, or labeling change. This active window should be distinct yet not distract the user's attention from window activity (see Figure 5-15).

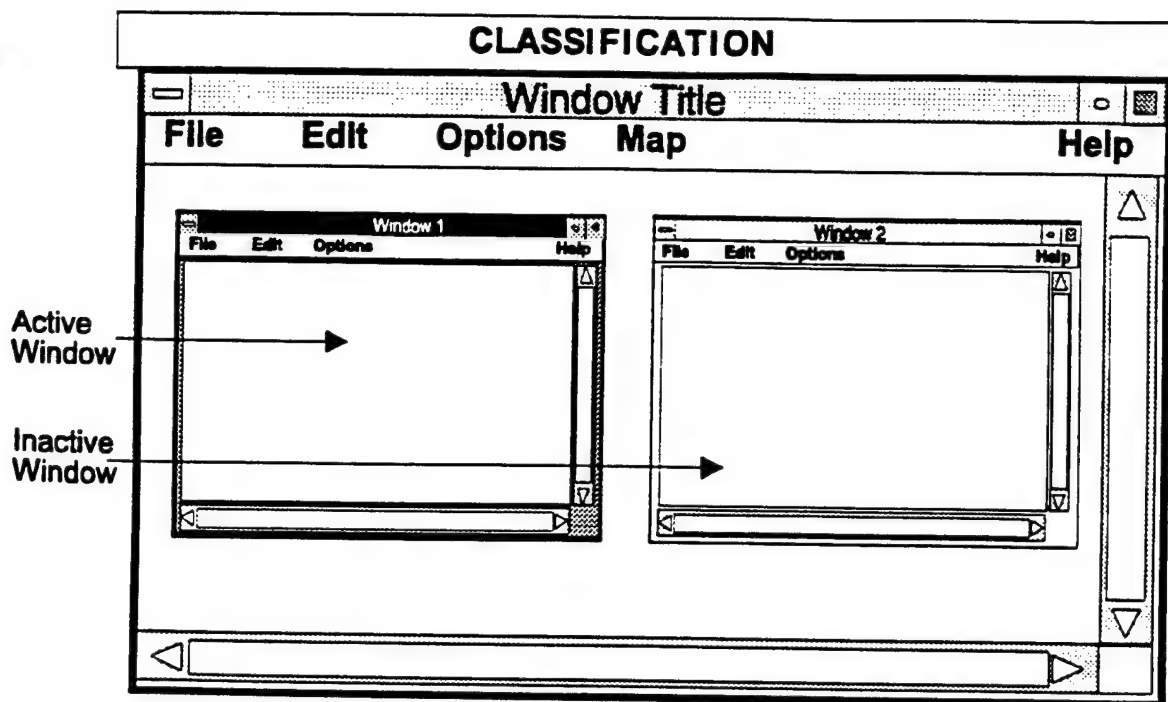
#### 5.2.3.2 Easy Shifting Among Windows

If several window objects are displayed at once, provide some easy means for the user to shift among them to select which window will be currently active. For example, shift the cursor with the mouse, then press the mouse button to designate the active window.

### 5.2.4 Labeling

#### 5.2.4.1 Labeling Windows

Assign an identifying label to window objects, dialog boxes, or subordinate windows. This label should briefly describe the contents, purpose of the window, or the menu path (e.g., Messages:email:outbox).



**Figure 5-15. Example of Active Window Designation**

#### **5.2.4.2 Format of Subordinate Window Labels**

Ensure that titles of subordinate windows match menu selection items from the supraordinate window menu.

#### **5.2.4.3 Window Titles**

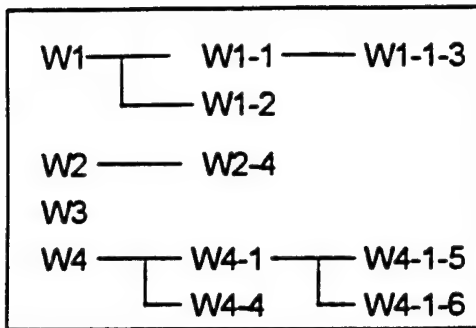
Locate window titles consistently.

#### **5.2.5 Open Window Navigation**

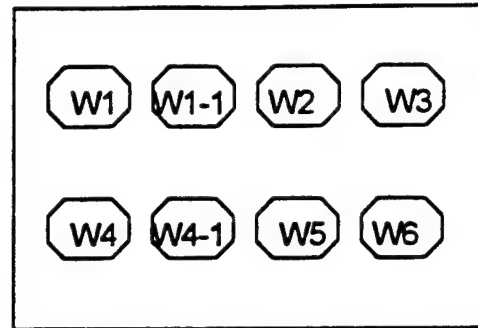
Navigation, in terms of windows, refers to the user's ability to move among the various windows that are open on a display.

##### **5.2.5.1 Open Window Map**

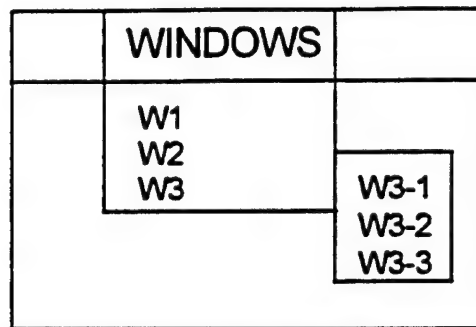
Applications should, when using an overlapping window structure, provide a user-requested iconic or text map/indication of all open windows to allow the user to easily identify all open (especially the hidden) windows. Figure 5-16 shows three presentations of an open window map.



Flowchart Presentation



Iconic Presentation



Pull-Down Window Presentation

**Figure 5-16. Examples of Open Window Maps**

#### 5.2.5.2 Active Designation from Open Window Map

Provide the user the capability to designate the active window through the iconic or text open window map by highlighting the window representation.

#### 5.2.5.3 Expanded Window Explanation of Open Window Map

If possible, allow the user to query an open window map for expanded information (e.g., date created, size, description of subject or application, etc.) on the file or application operating in the window. This information is usually accessed through HELP.

#### 5.2.5.4 Window Forward Function With Window Map

When an iconic or text map is provided for determining the numbers and names of open windows in an overlapping system, allow the user to bring a window forward from the map without having to resize or move other windows.



## REFERENCES

Paragraph	References
5.0	Avery and Bowser (1992); Billingsley (1988); DISA/CIM (1992)
5.1	DISA/CIM (1992)
5.2.1.1	Billingsley (1988) p. 416
5.2.1.2a	Avery et al. (1990); Smith and Mosier (1986) paras 2.5.2.5-3 and 2.5.8
5.2.1.2b	Avery et al. (1990); Smith and Mosier (1986) para 2.5.2.5-2
5.2.1.3a	Smith and Mosier (1986) para 2.5.2.5-1
5.2.1.3b	Slominski and Young (1988); Avery et al. (1990)
5.2.1.3c	Smith and Mosier (1986) para 2.5.2.5-10
5.2.1.4	Billingsley (1988) p. 421
5.2.1.5a	Avery et al. (1990); Billingsley (1988) p. 421; Bowser (1991) p 12; Smith and Mosier (1986) para 2.5-11; OSF (1990) Chapter 7
5.2.2.1a	Smith and Mosier (1986) para 2.5.2.5-9
5.2.2.1b	Smith and Mosier (1986) para 2.5.2.5-4
5.2.2.1c	Smith and Mosier (1986) para 2.5.2.5-5
5.2.2.1d	Nielson (1987) p. 247
5.2.2.2a	Billingsley (1988) p. 428
5.2.2.2b	OSF (1990) p. 3-4
5.2.2.3a	Billingsley (1988) p. 428
5.2.2.3b	Billingsley (1988) p. 428
5.2.2.4a	Slominski and Young (1988); Avery et al. (1990)
5.2.2.4b	Billingsley (1988) p. 429
5.2.2.4c	Billingsley (1988) p. 421; OSF (1990) p. 4-5; Avery et al. (1990)
5.2.2.4d	OSF (1990) p. 3-6
5.2.2.5a	Billingsley (1988) p. 430; OSF (1990)
5.2.2.5b	Billingsley (1988) p. 430
5.2.2.5c	Slominsky and Young (1988) p. 5; Avery et al. (1990)

## REFERENCES (cont'd)

Paragraph	References
5.2.3.1	Billingsley (1988) p. 431; Lewis and Fallesen (1989) p. 94; Smith and Mosier (1986) paras 2.5.2.5-7
5.2.3.2	Smith and Mosier (1986) para 2.5.2.5-8
5.2.4.1	Smith and Mosier (1986) para 2.5.2.5-6 and 2.5-10; OSF (1990) p. 3-4
5.2.4.2	Slominski and Young (1988) p. 2, 54 p. 3-4
5.2.4.3	Slominski and Young (1988) p. 2
5.2.5.1	Billingsley (1988) p. 420
5.2.5.2	Billingsley (1988) p. 431
5.2.5.3	Billingsley (1988) p. 420
5.2.5.4	Billingsley (1988) p. 420

## 6.0 MENU DESIGN

Using menus as a dialog is widespread within computer systems. Menus are frequently used in conjunction with other interactive methods, such as direct manipulation.

Using menus as a dialog has advantages, a major one being that it requires little training or sophistication on the part of the user. A user needs know only the meaning of each menu option, then is guided step by step through the operation of the system. The number of keystrokes required to access a system function may also be reduced, thereby speeding the user-to-computer transaction.

On the other hand, using menus as a dialog has disadvantages. It does not enhance retention of commands and may actually increase response time for the more experienced user. Menus may take up a large part of the display surface. In addition, for complex sequences, using menus may require an extensive menu tree structure, and the user may easily become lost navigating through a complex menu tree.

A number of different types of menuing techniques are available to the designer, including pull-down, pop-up, and sequential display. Pull-down and pop-up menus tend to be used more in direct manipulation types of dialog. Sequential display, where a control action causes another menu to overwrite the previous menu, is used more in text-based systems. All these types of menuing techniques can be hierarchical, or branching, in nature.

The following pages provide detailed design guidelines for menus used in operational systems. To ensure a high level of user performance with menus, the designer should be aware of the following general guidelines:

- Consider choosing menus when:
  - tasks involve choosing among a constrained set of alternative actions
  - tasks require infrequent entry of data
  - the user may have little training
  - the computer response is relatively fast
  - tasks require infrequently used commands
  - command sets are so large that the user is not likely to commit all commands to memory.
- Design the menu tree structure broad and shallow, rather than narrow and deep. Keep the number of top-level options large, with a small number of sublevels.
- Consider the experienced user and provide a mechanism by which the menu structure can be bypassed using a direct command.

The designer should also note that some of the guidelines discussed in the following paragraphs may be more appropriate for designing sequential display menus than for menus used in direct manipulation. The designer should use judgment regarding which approach to take.

The designer should conform to a single interface style, such as "Motif," throughout an application. Varying interface styles confuses the user. Widgets, or graphical objects that are components of a user interface, or graphics as menu item selectors should be unique and clearly identifiable by the user.

## **6.1 GENERAL**

### **6.1.1 Consider Response Time and Display Rate**

If computer response time is long, create menus with a larger number of items. If display rate is slow, create menus with fewer items to reduce display time.

### **6.1.2 Instructions and Error Messages**

Indent menu instructions and error messages and place them in the same position on the screen so the user knows where to look for this information.

### **6.1.3 Explicit Option Display**

When entries for any particular computer transaction consist of a small set of options, show those options in a menu added to the working display, rather than require a user to remember them or access a separate display.

### **6.1.4 Stacking Menu Selections**

For menu selection by code entry, when a series of selections can be anticipated before the menus are displayed, permit the user to combine those selections into a single stacked entry. Stacking refers to stringing multiple commands together and executing them with one action.

### **6.1.5 Menus Distinct From Other Displayed Information**

If menu options included in a display are also intended for data review and/or data entry, ensure they are distinct from other displayed information. Locate menu options consistently; use consistent visual cues for their special function.

### **6.1.6 Menu Bars**

Menu bars provide system functions in a bar across the top of the display screen. The following guidelines apply to menu bars.

### 6.1.6.1 Using Menu Bars

A menu bar is best used with standard sized screens (12-19 inches). With large-screen displays, the distance the pointer is required to travel may be too great to be effective. In this context, large-screen displays are defined as intended for multiple viewers, including projections and theater-type displays.

### 6.1.6.2 Visibility of Menu Bar Options

Ensure that menu bar options remain constantly visible (see Figure 6-1).

### 6.1.7 Pull-Down Menus

Pull-down menus, as illustrated in Figure 6-2, are lists of options attached to a selection on the menu bar that remain visible until the user takes action. Use pull-down menus instead of pop-up menus when pointer position on the screen is not important for information/option retrieval.

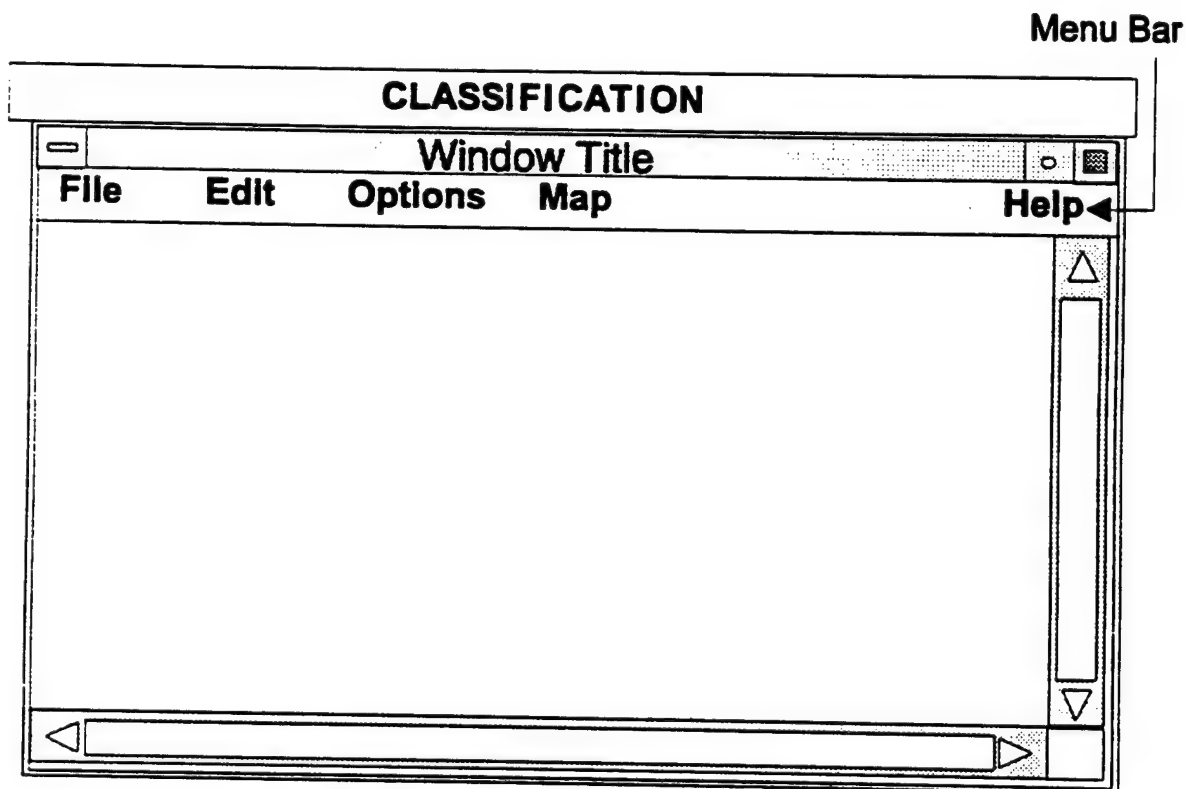
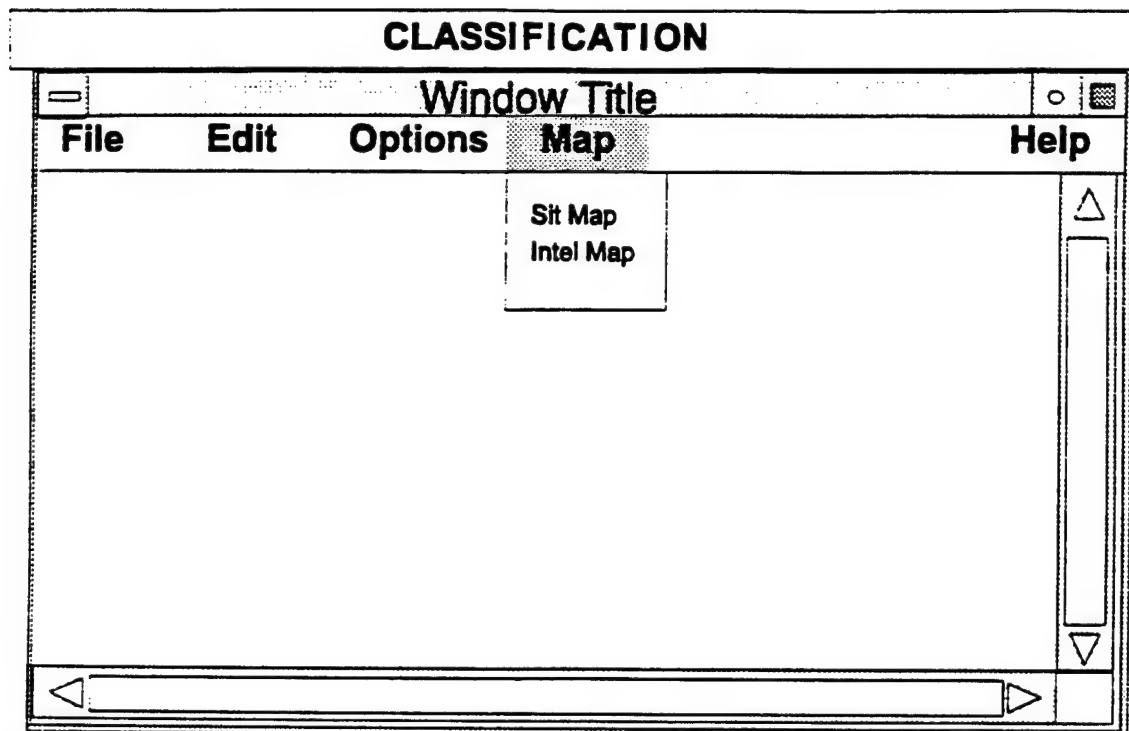


Figure 6-1. Example of Menu Bar



**Figure 6-2. Example of a Pull-Down Menu**

### **6.1.8 Pop-Up Menus**

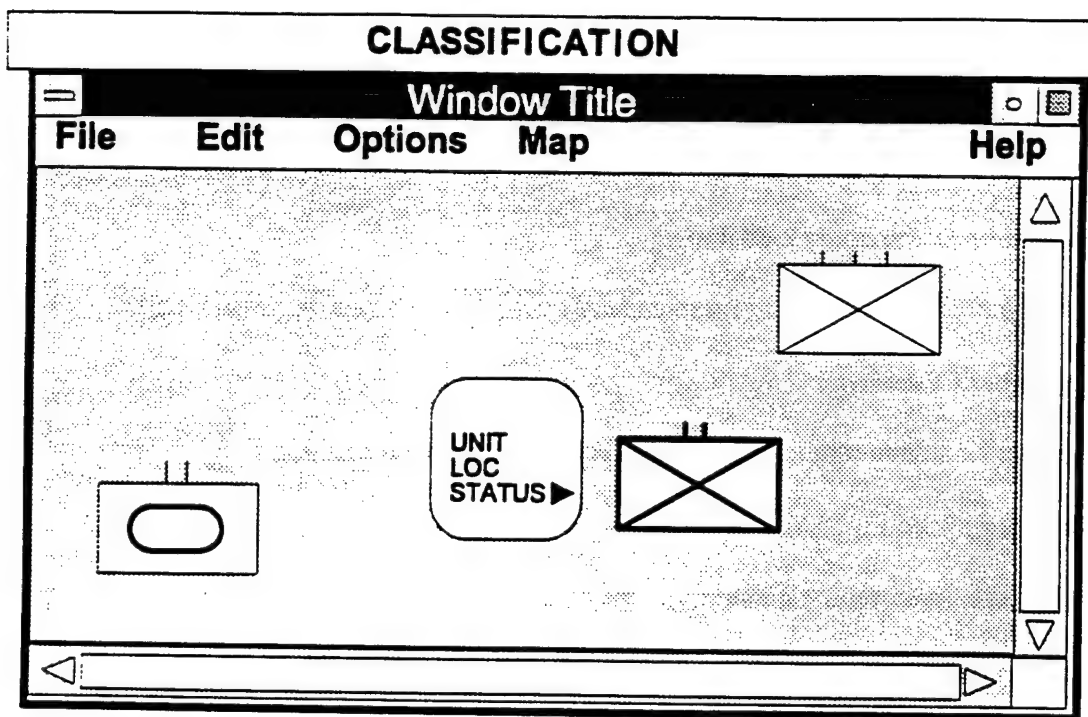
Pop-up menus are lists of options that appear on the display screen in the form of a window (see Section 5.0). Pop-up menus are specific to their area on the display; each window or object may have its own individual pop-up menu. The following guidelines should be used when designing pop-up menus.

#### **6.1.8.1 Pop-Up Menu Location**

Ensure that pop-up menus are connected to pointer location and pop up near the object or higher level menu being manipulated. See map overlay, Figure 6-3.

#### **6.1.8.2 Selecting Options From Pop-Up Menus**

Two methods to select from a pop-up menu are: 1) hold the button down while traversing options, then release to make the selection, or 2) move the pointer and press the button again for the selection. Use the second method when a choice is made to use only one selection method. Although it involves more keystrokes, it is less error-prone. It is acceptable to enable the application to allow either method and give the choice to the user.



**Figure 6-3. Example of a Pop-Up Menu**

### **6.1.8.3 Selection Highlighting**

When an option has been selected from a pop-up menu, ensure that it remains highlighted.

## **6.2 FORMAT**

### **6.2.1 General**

#### **6.2.1.1 Menu Format**

Keep lists of menu and submenu items brief (no more than five to nine options), arranged in separate columns, aligned, and left-justified.

#### **6.2.1.2 Consistent Display of Menu Options**

When menus are provided across different displays, design them so option lists are consistent in wording and order.

#### **6.2.1.3 Logical Grouping of Menu Options**

Format a menu to indicate logically related groups of options, rather than an undifferentiated string of alternatives.

#### 6.2.1.4 Logical Ordering of Grouped Options

If menu options are grouped in logical subunits, display those groups in a logical order. If no logical structure is apparent, display the groups in the order of their expected frequency of use. See the example in Figure 6-4.

#### 6.2.1.5 Sequence or Frequency Ordering

For a small number of menu items, use sequence or frequency to determine menu order.

#### 6.2.1.6 Alphabetic Ordering

For a large number of menu options, use alphabetic ordering of menu items.

#### 6.2.1.7 Numbering Menu Options

When task order is important, list menu options by number, not by letter.

#### 6.2.1.8 Display of Options

In designing a menu for a GUI, display unavailable menu items in a visually distinct manner. Refer to Paragraph 8.3.3.5.

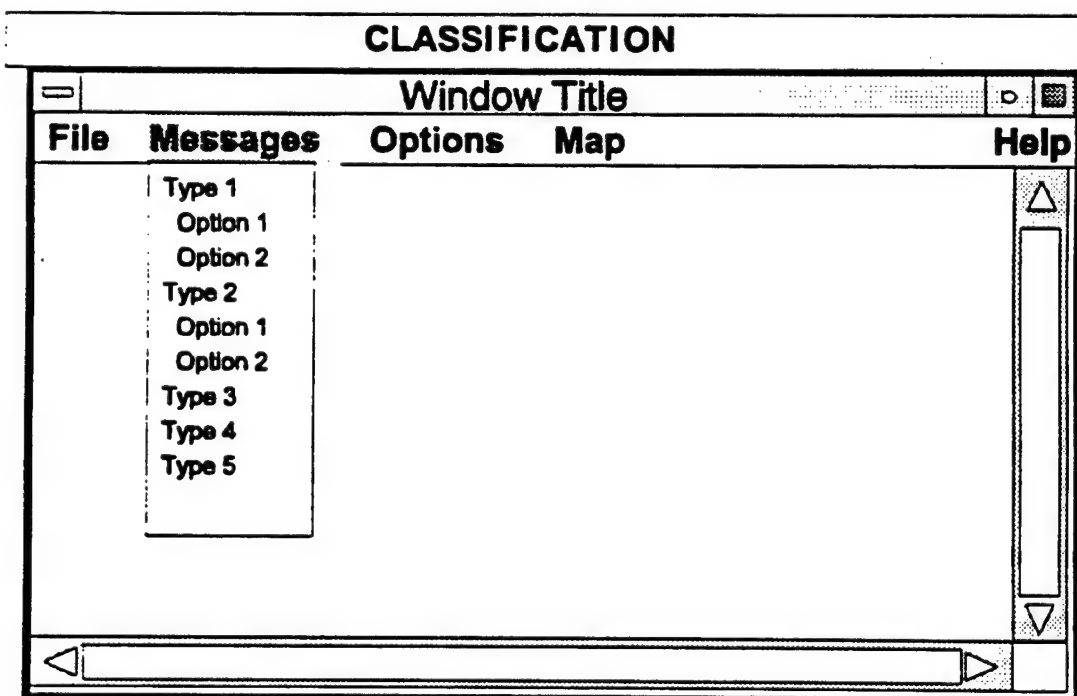


Figure 6-4. Example of Logical Ordering of Grouped Options



#### **6.2.1.9 Single-Column List Format**

When multiple menu options are displayed in a list, display each option on a new line (i.e., format the list as a single column).

#### **6.2.1.10 Overlapping Items**

Ensure that menu options do not overlap controlled functions or appear to do so to the user.

### **6.3 HIERARCHICAL MENUS**

#### **6.3.1 Usage**

Use hierarchical menus:

- When menu selection must be made from a long list and not all options can be displayed at once
- If a selection list exceeds 10-15 items.

#### **6.3.2 General Guidance**

##### **6.3.2.1 Organization and Labeling of Hierarchical Menus**

When hierarchical menus are used, organize and label them to guide the user within the hierarchical structure. Identify currently active menu selections to the user. The preferred method is to use more than one mode (i.e., color and font, size and color of text, etc.).

##### **6.3.2.2 Easy Selection of Important Options**

Design hierarchical menus to permit immediate user access to critical or frequently selected options.

##### **6.3.2.3 Indicating Current Position in Menu Structure**

When hierarchical menus are used, display an indication of the user's current position in the menu structure. This could be done in the menu title, or as a page X of N notation on the menu page.

##### **6.3.2.4 Consistent Design of Hierarchical Menus**

When hierarchical menus are used, ensure the display format and option selection logic are consistent at every level of the hierarchical menu structure.

### 6.3.2.5 Graphic User Interface for Hierarchical Menus

Keep hierarchical menu design in a GUI as simple as possible. The use of complex graphic structures is distracting to the user.

### 6.3.3 Navigating Hierarchical Menus

#### 6.3.3.1 Including a System-Level Menu

Provide a system-level menu of basic options as the top level in a hierarchical menu structure, as illustrated in Figure 6-5. The system-level menu will act as a home base to which a user can always return as a consistent starting point for control entries.

#### 6.3.3.2 Organization and Labeling System-Level Menu Listed Options

Group, label, and order control options for the system-level menu in terms of their logical function, frequency, and criticality of use.

#### 6.3.3.3 Return to the System-Level Menu

When hierarchical menus are used, require the user to take only one simple control action to return to the system-level menu.

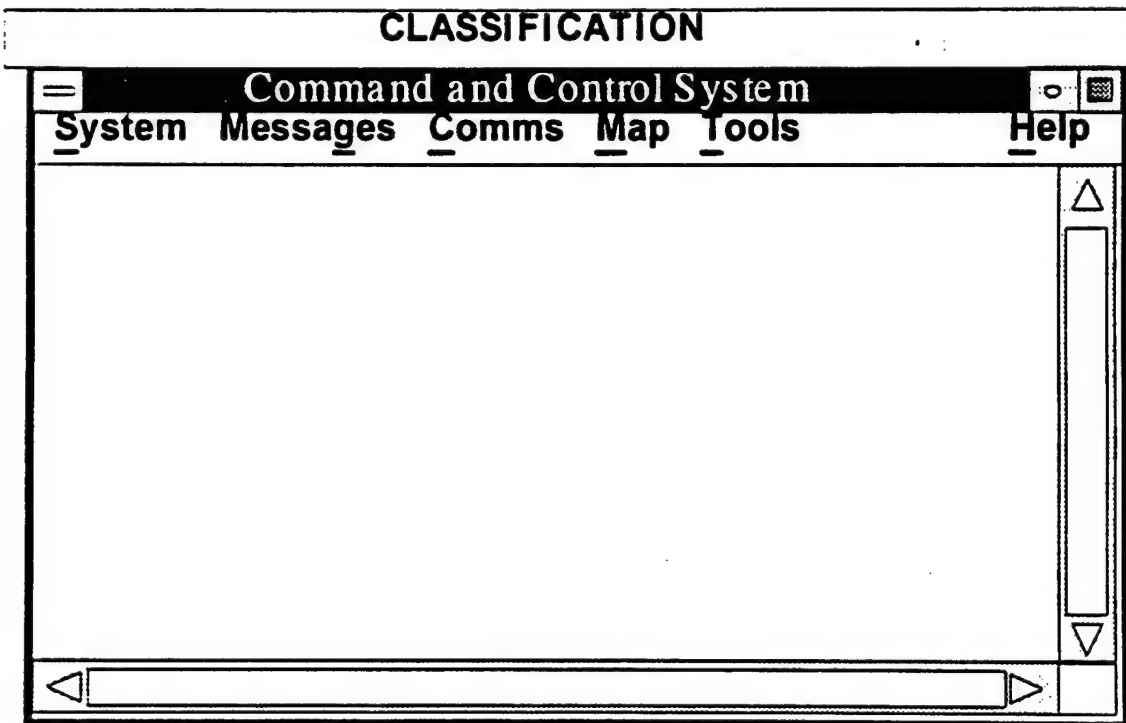


Figure 6-5. Example of a System-Level Menu

#### 6.3.3.4 Return to Higher Level Menus

When hierarchical menus are used, require the user to take only one simple control action to return to the next higher level.

#### 6.3.3.5 Control Options Distinct From Menu Branching

Format the display of hierarchical menus, dialog boxes, and pop-up windows such that options that actually accomplish control entries can be distinguished from those which merely branch to other menu frames. See Figure 6-6.

#### 6.3.3.6 Hierarchical Menu-Browsing Methods in Direct Manipulation

Two basic methods for browsing options in hierarchical menus are used in direct manipulation interactive control: 1) select an option from one menu, which causes another menu to pop up, or 2) move the pointer towards the right side of an option, causing a menu to pop up (see Figure 6-7). It is recommended that both options be available.

#### 6.3.3.7 Use of Multiple Paths

Provide multiple paths to accommodate both the experienced and inexperienced user. Allow the experienced user to use "type-ahead," "jump-ahead," or other shortcuts to navigate through the menu selection system.

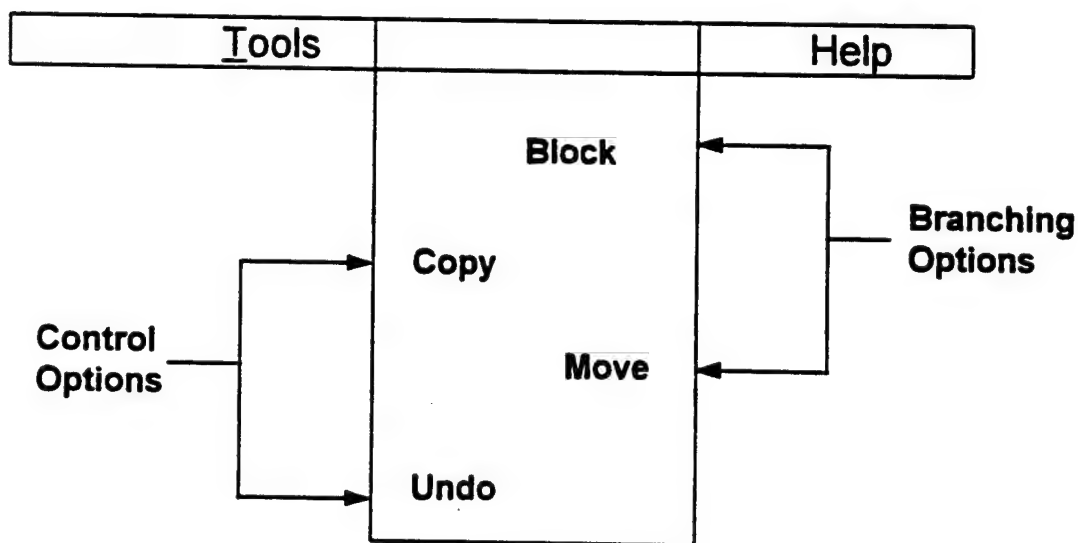
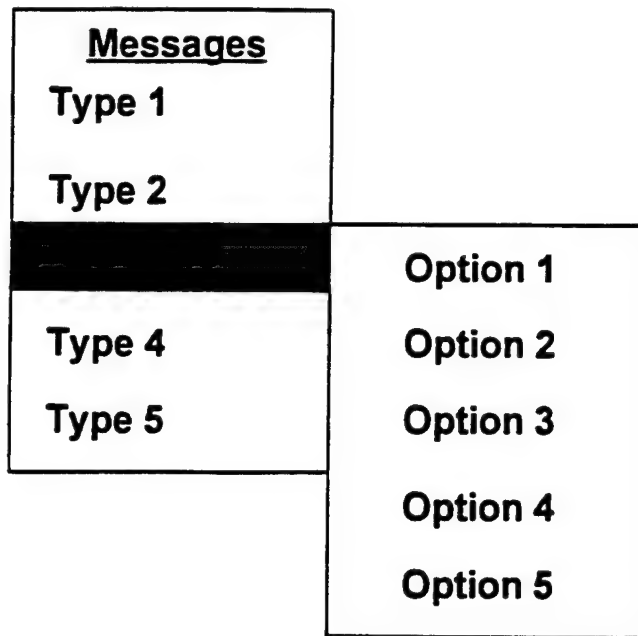


Figure 6-6. Distinction Between Control Options and Command Options



**Figure 6-7. Example of a Hierarchical Menu**

#### **6.3.4 Hierarchical Menu Tree Depth and Breadth**

##### **6.3.4.1 Minimal Steps in Sequential Menu Selection**

When the user must step through a sequence of menus to make a selection, design the hierarchical menu structure to minimize the number of steps required.

##### **6.3.4.2 Use Broad Menu Trees**

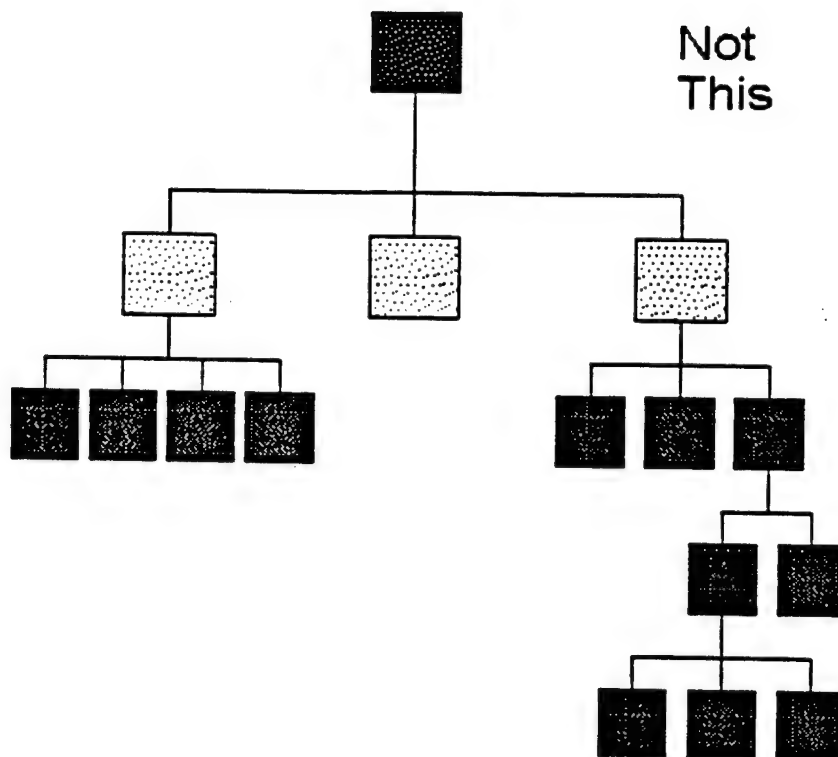
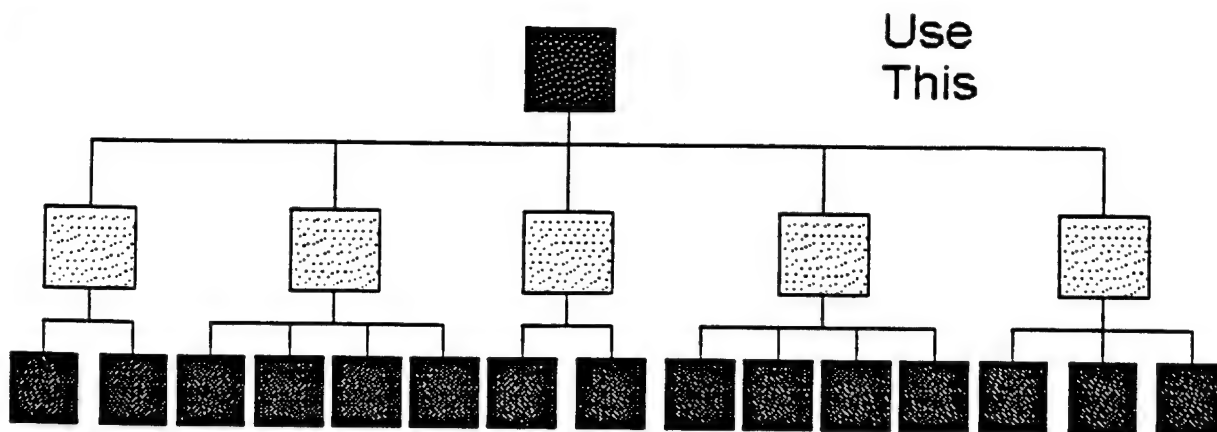
Use a broad and shallow menu tree, rather than a narrow and deep menu tree, for operational systems as illustrated in Figure 6-8.

##### **6.3.4.3 Minimize Menu Choices in the Middle**

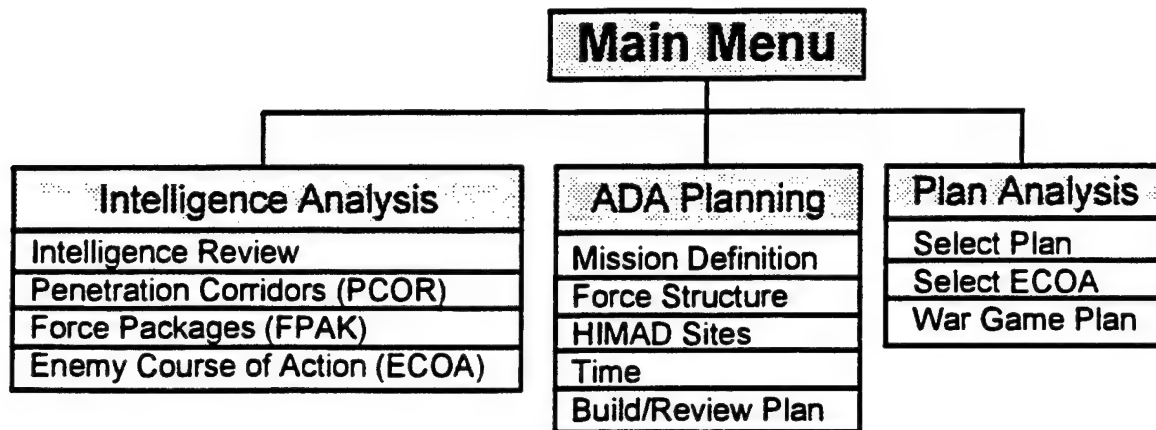
Minimize the number of menu choices midway through a hierarchical menu, as the user is more likely to get lost at this stage.

##### **6.3.4.4 Software Navigation Aids**

Include in software navigation aids the ability to select a menu or submenu directly, without going through intermediate steps (see Figure 6-9). Enable the user to switch between software modules in a quick, easy manner, using an interface such as a tree or organization chart.



**Figure 6-8. Broad and Shallow Menu Tree vs. Narrow and Deep Menu Tree**



**Figure 6-9. Example of a Tree Diagram Interface**

## **6.4 ITEM SELECTION**

### **6.4.1 General**

#### **6.4.1.1 Automatic Pointer Placement**

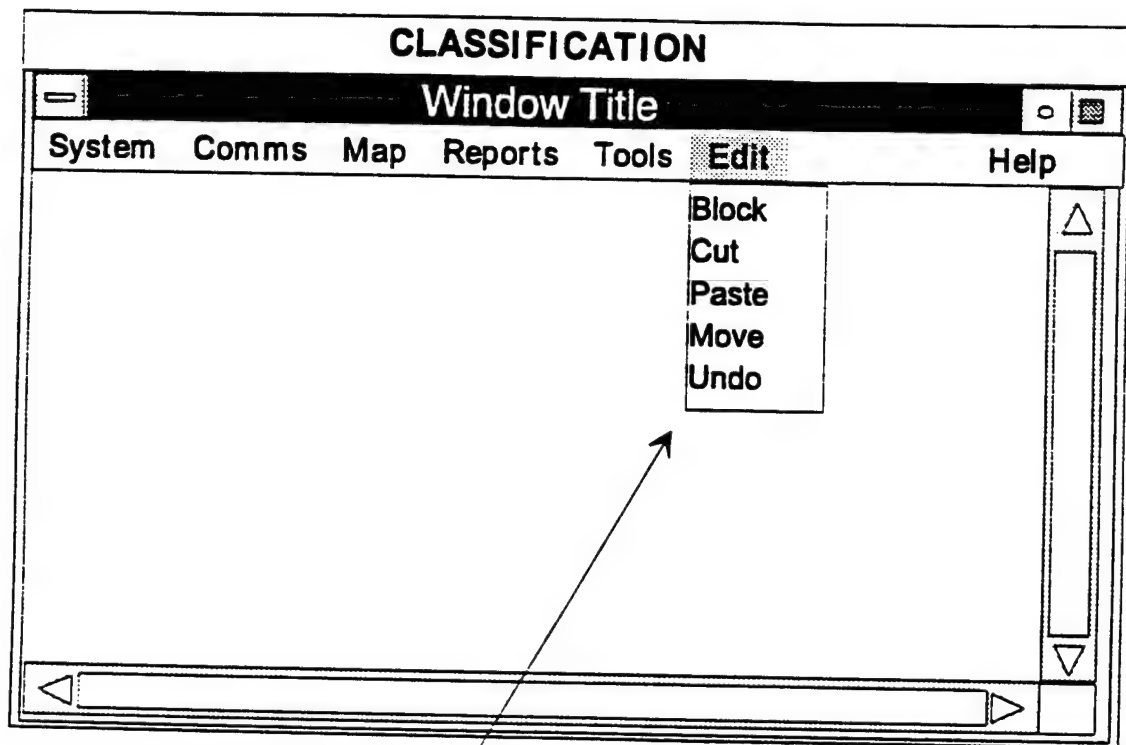
When pointing to make a menu selection on menu displays not included with data displays, ensure that the computer places the pointer automatically at the first listed option. When menu selection is by code entry, place the pointer in the command entry area.

#### **6.4.1.2 Minimize Menu Selections**

Keep the number of menu selections to the absolute minimum to reduce system menu-selection time.

#### **6.4.1.3 Use a Combined Mode of User Interface**

Enable users to use two modes for menu selection: keying in a numeric or letter code, or placing the pointer at the option and selecting (see Figure 6-10).



User Can:

- ① Move pointer to highlight option and select, or
- ② Type key letter code

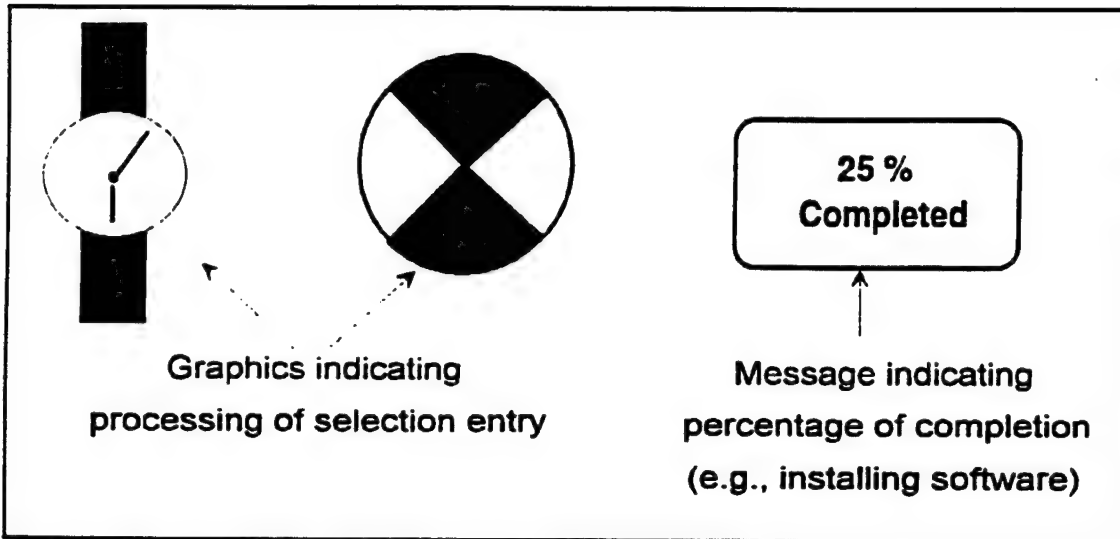
**Figure 6-10. Example of a Combined Mode User Interface**

#### **6.4.1.4 Feedback for Menu Selection**

When a user selects and enters a control option from a menu, if no natural response is immediately observable, the software should display some other acknowledgment of that entry. See examples in Figure 6-11. Where possible, the acknowledgment should be animated.

#### **6.4.1.5 Standard Area for Code Entry**

When menu selection is accomplished by code entry (other than mnemonics), provide a standard command entry area where the user enters the selected code. Place that entry area in a fixed location on all displays.



**Figure 6-11. Graphic Acknowledgments of Selection-Processing**

#### **6.4.1.6 Allow Abbreviated Menu Selections**

Allow menu selections by the user to be accepted in either abbreviated or complete form. For example, the user should be able to use Q, QU, or QUIT.

### **6.4.2 Selection By Pointing**

#### **6.4.2.1 Menu Selection by Pointing**

If menu selection is the primary means of sequence control, and especially if choices must be made from extensive lists of displayed control options, permit option selection by direct pointing (e.g., mouse, trackball). See Subsection 7.1, general aspects of direct manipulation.

#### **6.4.2.2 Large Pointing Area for Selecting Options**

The acceptable pointing area for menu options should be as large as is consistently possible. Ensure that the area includes at least the displayed option label, plus a half-character distance around that label.

#### **6.4.2.3 Dual Activation for Pointing**

If pointing for menu selection, provide dual activation, where the first action designates (positions a cursor at) the selected option and a separate, second action makes an explicit control entry (e.g., clicking the mouse).



## **6.5 MENU OPTION LABELING**

### **6.5.1 General**

#### **6.5.1.1 Use of Key Words**

Ensure that menu items begin with a key word.

#### **6.5.1.2 Menu Options Worded as Commands**

Ensure that the wording of menu options consistently represents commands to the computer (e.g., File, Save, Edit), rather than questions to the user.

#### **6.5.1.3 Menu Categories**

Ensure that menu category labels are comprehensible and unique. The words, phrases, and titles should state options in clear English.

#### **6.5.1.4 Labeling Grouped Options**

If menu options are grouped in logical subunits, give each subunit a descriptive label distinctive in format from the option labels themselves (see Figure 6-12).

#### **6.5.1.5 Use Familiar Terminology**

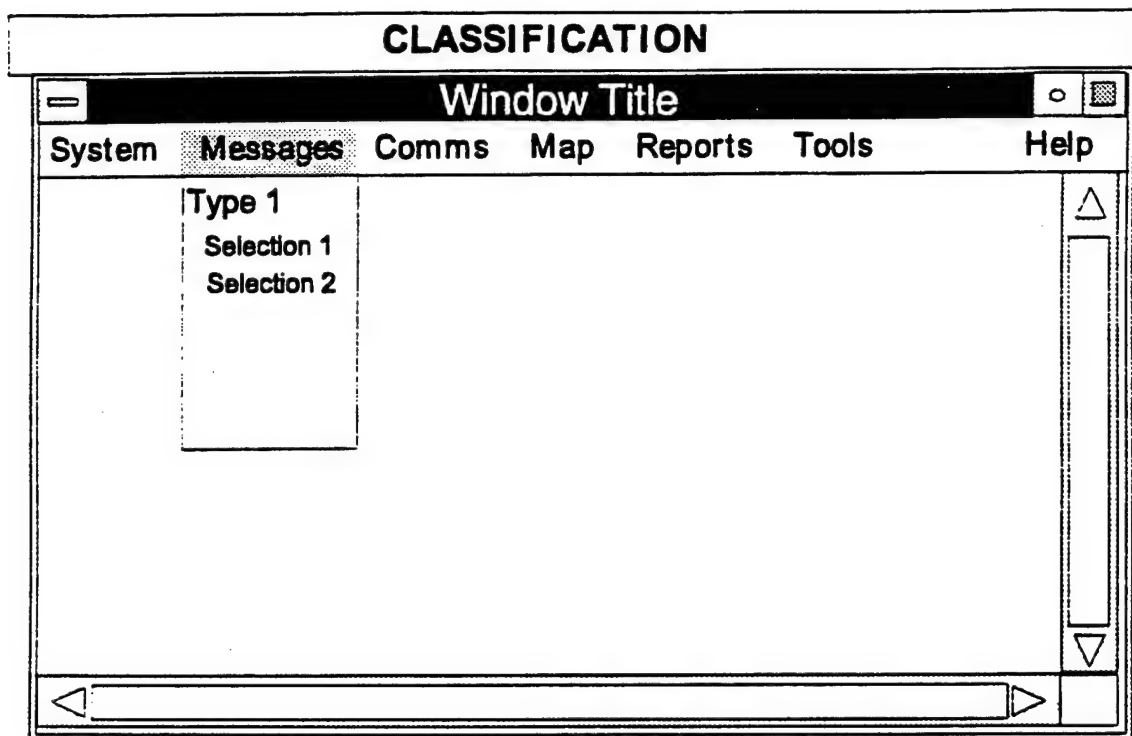
Use familiar terminology when labeling menus, but ensure that items are distinct from one another.

### **6.5.2 Selector**

#### **6.5.2.1 Best and Worst Selectors for Menu Items**

Mnemonics is a technique to assist in improving the user's memory. Compatible or mnemonic letters are the best selectors for menu items; incompatible letters are the worst. Numbers are intermediate selectors.

- Use lettered menu items if possible, as they have the following advantages: more single entry keys are available; there is less chance of a keying error; and mnemonic keying of entries is possible.
- Use numbered menu items as intermediate selectors, with the following advantages: sequencing of items is clear; non-typists can easily locate numbers; and the user can quickly see how many options are available.



**Figure 6-12. Example of Distinctive Subunit Labels**

#### **6.5.2.2 Do Not Combine Codes**

Letter and numeric codes should not be combined in the dialog.

#### **6.5.2.3 Selection of Menu Titles**

Use selectors that closely match the item represented, to facilitate user retention of commands.

#### **6.5.2.4 Numbering**

Number menu items starting with 1 -- not with 0.

#### **6.5.2.5 Consistent Coding of Menu Options**

If letter codes are used for menu selection, use those letters consistently in designating options from one transaction to another.

#### **6.5.2.6 Displaying Option Code**

When the user must select options by code entry, display the code associated with each option in a consistent, distinctive manner, as shown below.

Code		Option
P	=	Previous Page
N	=	Next Page
U	=	Undo
Del	=	Delete

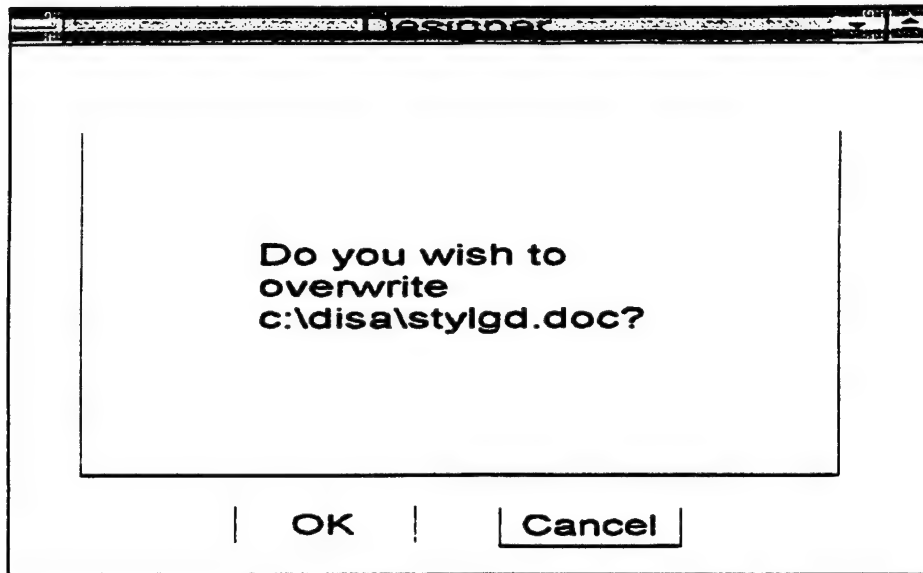
## 6.6 DIALOG BOXES/POP-UP WINDOWS

GUI style guides refer to windows that contain graphical controls (widgets), such as dialog boxes and pop-up windows (see Subsection 5.1, Window Basics), for interacting with applications. Examples of dialog boxes include message, question, warning, action, and command windows; examples of pop-up windows include command windows, property windows, and notices. Note that dialog box and pop-up command windows are not equivalent or even related. The former refers to a window that allows users to enter commands to the application or operating system, and the latter is a window that sets parameters and executes commands based on those parameters. These windows are used to:

- Display important messages or warnings
- Collect or solicit data from the user
- Modify and set properties of objects
- Notify the user of the progress of a lengthy process.

Dialog boxes and pop-up windows are invoked by applications in response to 1) user actions and requests, 2) unexpected or unplanned events (e.g., a printer runs out of paper), or 3) initiation of a time-consuming activity. See the dialog box in Figure 6-13. The application decides where and when they are displayed, but all dialog boxes and pop-up windows should include at least one button that solicits a response from the user. Windows should be noticeable but small and, if possible, moveable. It is recommended that only one dialog box or pop-up window be displayed at a time within any application in order to avoid clutter and confusion.

Dialog boxes and pop-up windows should automatically receive input focus. Users should be required to respond to dialog boxes or pop-up windows and should be prevented from returning the input focus to the main or primary window (of the application) until they have responded appropriately.



**Figure 6-13. Example of a Dialog Box**

All types of dialog boxes and pop-up windows behave similarly, but they differ in content depending on the needs of the application. For example, a push button is always pushed and a check button is always checked, but each application will choose the types of controls to use and combine them differently. The following paragraphs provide recommendations for message wording and briefly describe some common types of dialog boxes and pop-up windows. More detailed descriptions can be found in the OAF/Motif and Open Look style guides.

#### **6.6.1 Message Wording Guidelines**

The following guidelines, which are designed to maximize user performance and accuracy, should be applied to dialog boxes, pop-up windows, message areas, and any other communications between the application and user.

- Use an abbreviation only when it is significantly shorter than the full word.
- Ensure that abbreviations are meaningful, recognizable, and used consistently.
- Do not abbreviate words not commonly abbreviated. For example, use "Restricted Acct No," not "Restr Account Number."
- Ensure that message lines end in full words rather than in hyphenations.
- Ensure that messages are directly usable, requiring no further documentation or translation.
- Avoid overly technical wording, and use short simple sentences that begin with the main topic.

- Avoid abrupt wording, such as INVALID, ILLEGAL, and FATAL.
- Focus error messages on the procedure for correcting the error, not on the action that caused the error.
- Display critical error messages (those requiring immediate response from the user to prevent invalid data or results) in caution/warning windows, as shown in Paragraph 6.6.4. Display noncritical messages in the message area at the bottom of the application window, as described in Paragraph 5.1.5.
- Where appropriate, the HELP facility can be used to expand more fully on messages.

### **6.6.2 Work-In-Progress Window**

When a user's request is simple and requires five seconds or less processing time, feedback can be in the form of a changed pointer shape or a brief message within the window. When the request exceeds five seconds, the application should provide a work-in-progress window to indicate a time-consuming operation is taking place. If appropriate, provide a means by which the operation can be canceled or aborted. The application removes the box when the operation has been completed.

Ensure that the application shows the status of the operation by a dynamically changing progress indicator (e.g., "10% Sorted," "4 out of 10 files copied," or a scale showing status).

### **6.6.3 Information Box**

An application should generate an information box (i.e., a Motif message box or an Open Look notice) when the application needs to display an information message. This window should be reserved for noncritical messages requiring acknowledgment by the user. An application's frequent informational messages should be displayed in the window's message area (see Paragraph 5.1.5).

An information box can freeze the application and require the user to explicitly dismiss the window before proceeding. If the halted operation can be retried, include a "Retry" button within the message window. If a default push button is designated, assume it is the desired action.

### **6.6.4 Caution/Warning Box**

A caution/warning box, containing critical messages that warn the user of the consequences of carrying out an action, usually includes "Yes," "No," and Cancel" buttons. The message should be an unambiguous question or statement. When this box is displayed, suspend the application until the user provides instructions on how to proceed. Ensure that the default push button is always the least destructive operation.

### **6.6.5 Menu Box**

A menu box is the result of the user's selecting a routing or window menu item. Menu boxes solicit data from users through a combination of controls (e.g., entry boxes and settings). Name the menu box in accordance with the menu item that created it. For example, the "search..." menu item should generate a menu with the title "Search...." A "Cancel" push button should be included in the window to allow users to close the menu box. If a default push button is designated, it should be the assumed desired action.

## REFERENCES

Paragraph	References
6.1.1	Shneiderman (1987) p. 107
6.1.2	Shneiderman (1987) p. 115
6.1.3	Smith and Mosier (1986) para 3.1.3-16; Sidorsky (1984) para 6.1-14
6.1.4	Smith and Mosier (1986) para 3.1.3-36
6.1.5	Smith and Mosier (1986) para 3.1.3-20
6.1.6.1	Ziegler and Fährnich (1988) p. 130
6.1.6.2	Ziegler and Fährnich (1988) p. 130
6.1.7	Ziegler and Fährnich (1988) p. 130
6.1.8.1	Ziegler and Fährnich (1988) p. 129
6.1.8.2	Ziegler and Fährnich (1988) p. 129
6.1.8.3	Ziegler and Fährnich (1988) p. 129
6.2.1.1	Lickteig (1989) p. 13, 30 Appendix A p. A-1; Bailey (1982) p. 346; Chao (1986) p. 15
6.2.1.2	Smith and Mosier (1986) para 3.1.3-19; Shneiderman (1988) p. 702.
6.2.1.3	Smith and Mosier (1986) para 3.1.3-22 and 4.4-3; Shneiderman (1988) p.702; Paap and Roske-Hofstrand (1986) p. 384
6.2.1.4	Smith and Mosier (1986) para 3.1.3-23; Lickteig (1989) p. 15; Shneiderman (1988) p. 702
6.2.1.5	Galitz (1984) p. 120
6.2.1.6	Galitz (1984) p. 120; Chao (1986) p. 16
6.2.1.7	Williams et al. (1987a) Appendix p. A-3
6.2.1.8	Smith and Mosier (1986) para 3.1.3-17 and 3.1.3-18; Sidorsky (1984) para 6.1-13
6.2.1.9	Smith and Mosier (1986) para 3.1.3-3
6.2.1.10	Shneiderman (1987) p. 100
6.3.1a	Smith and Mosier (1986) para 3.1.3-25; Nielsen (1987) p. 384; Lickteig (1989) p. 13; Ziegler and Fährnich (1988) p. 129

## REFERENCES (Cont.)

Paragraph	References
6.3.1b	Sidorsky (1984) para 6.1-14; Chao (1986) p. 15
6.3.2.1	Bowser (1991) p. 9; Smith and Mosier (1986) para 4.4-4
6.3.2.2	Smith and Mosier (1986) para 3.1.3-28; Sidorsky (1984) para 5.1-14; Galitz (1984) p. 119
6.3.2.3	Smith and Mosier (1986) para 3.1.3-30; DoD (1989a) p. 267; Chao (1986) p. 15; Shneiderman (1987) p. 115
6.3.2.4	Smith and Mosier (1986) para 3.1.3-32
6.3.2.5	Bowser (1991) p. 9
6.3.3.1	Smith and Mosier (1986) para 3.1.3-26, 4.4-2, 3.2-2; Shneiderman (1988) p. 702; Galitz (1984) p. 120; Sidorsky (1984) para 1.1-12, 3.2-21
6.3.3.2	Smith and Mosier (1986) para 3.2-3
6.3.3.3	Smith and Mosier (1986) para 3.1.3-34; Galitz (1984) p. 120; DoD (1989a) p. 266
6.3.3.4	Bowser (1991) p. 10; Smith and Mosier (1986) para 3.1.3-33; Sidorsky (1984) para 5.1-14; DoD (1989a) p. 267; Chao (1986) p.16
6.3.3.5	Smith and Mosier (1986) para 3.1.3-31; Galitz (1984) p. 120
6.3.3.6	Ziegler and Fähnrich (1988) p. 129
6.3.3.7	Smith and Mosier (1986) para 3.1.3-35; Shneiderman (1988) p.702; Shneiderman (1987) p. 118; Chao (1986) p. 16; Laverson and Shneiderman (1987) p. 104; Sidorsky (1984) para 5.1-15
6.3.4.1	Smith and Mosier (1986) para 3.1.3-27; Sidorsky (1984) para 5.1-14
6.3.4.2	Shneiderman (1988) p. 702; Norman and Chin (1988) p. 63
6.3.4.3	Norman and Chin (1988) p. 63
6.3.4.4	Galitz (1984) p. 120
6.4.1.1	Galitz (1984) para 3.1.3-29; Chao (1986) p. 15
6.4.1.2	Parkinson et al. (1988) p. 691
6.4.1.3	Antin (1988) p. 181



## REFERENCES (Cont.)

Paragraph	References
6.4.1.4	Smith and Mosier (1986) para 3.1.3-9
6.4.1.5	Smith and Mosier (1986) para 3.1.3-8; Chao (1986) p. 15
6.4.1.6	Sidorsky (1984) para 6.1-15
6.4.2.1	Smith and Mosier (1986) para 3.1.3-4; DoD (1989a) p. 266
6.4.2.2	Smith and Mosier (1986) para 3.1.3-5
6.4.2.3	Smith and Mosier (1986) para 3.1.3-6
6.5.1.1	Shneiderman (1988) p. 702
6.5.1.2	Smith and Mosier (1986) para 3.1.3-11; Sidorsky (1984) para 5.1-13
6.5.1.3	Shneiderman (1987) p. 87; Bailey (1982) p. 343
6.5.1.4	Smith and Mosier (1986) para 3.1.3-24; Sidorsky (1984) para 5.1-14; Chao (1986) p.16
6.5.1.5	Shneiderman (1987) p. 100
6.5.2.1	Laverson et al. (1987) p. 106; Shneiderman (1987) p. 117 and 118; Chao (1986) p. 16; Smith and Mosier (1986) para 3.1.3-13; Galitz (1984) p. 120
6.5.2.2	Chao (1986) p. 16
6.5.2.3	Laverson et al. (1987) p. 105
6.5.2.4	Sidorsky (1984) para 2.1-13; Chao (1986) p. 16
6.5.2.5	Smith and Mosier (1986) para 3.1.3-14; Sidorsky para 5.1-14
6.5.2.6	Smith and Mosier (1986) para 3.2-8; Shneiderman (1987) p. 115
6.6	DISA/CIM (1992b)

**This page intentionally left blank.**

## 7.0 DIRECT MANIPULATION

Direct manipulation is the major type of interactive dialog for GUIs. In a direct manipulation dialog, the user controls the interface with the computer by acting directly on "objects" on the display screen. This object may be an icon, menu option, symbol, button, or dialog box. The user highlights the object and implements the action by using a pointing device, such as a mouse or trackball. Sample actions include moving an object, querying a database, calling up a preformatted message template, or sending a message over a communications system. The result of an action is immediately observable.

Direct manipulation of a computer system is analogous to controlling a vehicle. The user uses a control, such as the steering wheel, to input a command to the vehicle and is rewarded by an immediate response. With a computer, the user moves a pointer over an object, such as an icon, and presses a pointing device control (i.e., button) to input a command, such as querying status. The direct manipulation system responds immediately by displaying the status in a pop-up window next to the object. In contrast, when using a command-language-based system, the user types in a command, hits ENTER, then waits for a response from the system.

Direct manipulation user interfaces are characterized by continuous representation of the object of interest and by computer actions accomplished by physical actions such as button presses, incremental reversible actions, and immediate visual feedback. These characteristics provide the user with a greater feeling of control and often result in better performance and greater acceptance of the system.

Direct manipulation in the user interface reduces the time required to learn new applications. Efficiencies in learning result from using both standard, consistent actions in the application environment and metaphors to guide the user. A metaphor uses the visual nature of a direct manipulation interface to map objects in the application onto a visual representation familiar to the user.

Metaphors effectively control the complexity of the user interface because they make actions, procedures, and concepts similar to those already known to the user. By capitalizing on the user's prior knowledge, the designer permits the user to think in terms familiar to the application domain, rather than in terms of low-level computer concepts. The resulting applications are easier to learn and easier to use. An effective example is the office metaphor associated with the Apple-MacIntosh interface style.

The following pages provide detailed guidelines for designing the user-computer interface when direct manipulation is used. The designer should recognize that the literature has little explicit guidance for direct manipulation as it applies to specific military systems. Therefore, it is imperative that before formalizing the user-computer interface, detailed research and careful testing of alternatives be done with users who represent the intended user population.

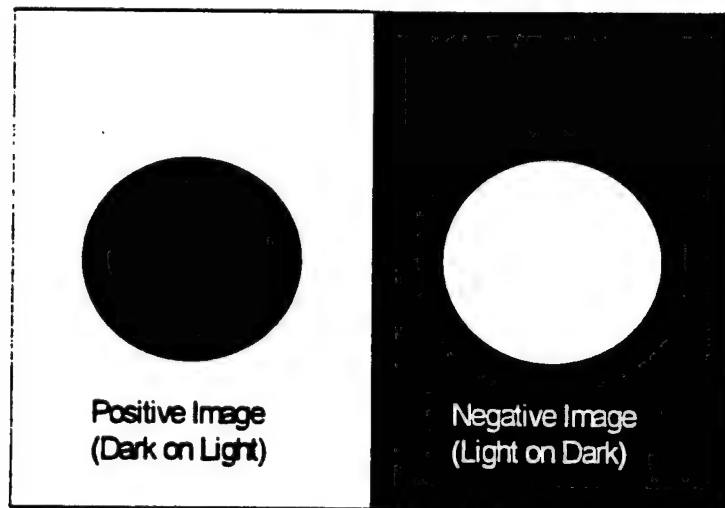
## 7.1 GENERAL

Provide direct manipulation of displayed objects as a means of interactive control. Direct manipulation works particularly well for applications where there will be many casual system users and where turnover in personnel will be high, such as in operational military situations.

### 7.1.1 Hardware Considerations

The designer should consider the following hardware factors for an effective direct manipulation system:

- Use high-resolution screens and a bitmapped hardware architecture, as these are required for direct manipulation systems. The bitmapped windowing system requires greater central processing and memory size as well as rapid operating speed to provide the immediate response for effective user-computer interaction.
- Because direct manipulation is designed to represent the actual product, a positive image (dark foreground on light background) is best, as it represents printed output. See the example in Figure 7-1.
- Direct manipulation is most efficient when using a pointing device, such as a mouse, trackball, or touch-interactive device. Section 3.0 discusses touch-interactive devices. Software must be flexible enough to accommodate keyboard cursor keys or accelerators should the pointing device fail.



**Figure 7-1. Example of Positive and Negative Images**

### **7.1.2 Screen Arrangement by the User**

Enable the user to arrange windows and icons on the screen to meet the individual task needs. However, do not allow the user to move a window or icon to a nonretrievable position (i.e., off the screen).

### **7.1.3 Function Control**

Five methods should be considered for invoking a function, file, or operation with direct manipulation: Function Keys, Menu Bar, Pop-up Menus, Pull-down Menus, and Icons. Icons are discussed in Subsection 7.3, function keys are discussed in Subsection 8.4, and menuing is discussed in Section 6.0.

### **7.1.4 Interaction**

Operator interactive tasks should use the most appropriate input mode. The keyboard is recommended for extensive alphanumeric data. It is usually more effective to use pointing devices to select from menus. Where both modes are present, the software should allow both keyboard and pointing device selection of items. Operational military systems should provide complete inter-changeability of keyboard and pointing device for emergency situations.

## **7.2 METAPHORS**

Metaphors associate interface objects in the application or functionality with a visual representation familiar to the user. To capitalize on information-carrying capacity, use metaphors to unify individual icons into integrated groupings using established attributes and associations of real-world objects. Icons replace commands and menus as the means to support end-user dialog (e.g., trash can replaces the delete command or menu item).

In investigating the range of understandability of symbols, research indicates the need to evaluate symbols and icons before their widespread adoption. The metaphor used for icons and system interaction should be tested in advance with representatives of the intended user population. Some symbols have very little meaning, can be misleading, and can cause potentially dangerous confusion in international environments.

Each society has unique meanings for different types of graphics and icons. These meanings have been developed in the cultural evolution process by associating objects in natural and man-made environments with life events and activities within the society. Therefore, icons and graphics of each culture can and should be studied not only intraculturally but also cross-culturally. Intercultural learning can be facilitated through formal education and cultural assimilation training of the semantic features involved in verbal and nonverbal communication.

## **7.2.1 Metaphor Selection**

Understandability remains of primary concern in achieving effective and widely-accepted symbols. The following paragraphs provide guidance on metaphor selection and design.

### **7.2.1.1 System Model**

The metaphor selected for icon design should model the system being controlled.

### **7.2.1.2 Appropriate to Task**

The metaphor selected for icon design should be appropriate for the user's tasks, functions, and environment (e.g., the office metaphor may not be appropriate for some military applications).

### **7.2.1.3 Leveraging Knowledge**

Selecting the metaphor should leverage prior knowledge in a way that is specific to the user environment.

### **7.2.1.4 Functional Representation**

The metaphor should represent the system function in a way that is meaningful to the user.

### **7.2.1.5 Generalization of Metaphors**

Metaphors should be general enough to allow the user to understand and use other metaphors or media, such as text-based systems.

## **7.2.2 Metaphor Design**

### **7.2.2.1 Complex Metaphors**

Avoid using complex metaphors. Complex metaphors, like complex icons, can lead to increased inferences of meaning and errors by the user. For example, the metaphor of biological evolution, if used to describe the levels and layers of an application, would be an overly complex metaphor.

### **7.2.2.2 Metaphor Oversimplification**

Although metaphors should be as simple as possible, avoid oversimplification. Oversimplification occurs when the metaphor does not model full capability of the system. This oversimplification can cause underutilization of the system functionality. For example, the metaphor of a notepad, if used to describe the levels and layers of an application, would be an overly simple metaphor.

### **7.2.2.3 Metaphor Consistency with Objects**

Metaphors should be consistent with the objects chosen to represent the functions. For example, deleting a file with recovery capability would be represented by a trash can, whereas deleting a file permanently would be represented by a paper shredder.

### **7.2.2.4 Metaphors Versus Self-Contained Icons**

If effective self-contained symbols (icons) can be designed for information presentation, use them over the multiple icons of a complex metaphor.

### **7.2.2.5 Metaphor Tutoring**

Design icon metaphors to tutor the user towards a more complete understanding of the underlying functional system.

### **7.2.2.6 Connotations Induced by Metaphors**

Develop metaphors carefully, especially those used by more than one cultural or national group (e.g., NATO forces). Ensure that metaphors do not have a negative connotation for the user. For example, the "OK" sign formed by touching the forefinger tip to the thumb tip carries obscene connotations for some cultures.

## **7.3 ICONS**

Icons are pictographic symbols representing underlying objects, concepts, processes, or data in a computer system. Icons are visible manifestations of a metaphor. Basic principles for designing icon and symbol systems are similar to those for designing large-scale windows and screens. Consistency, clarity, simplicity, and familiarity are key attributes. Sometimes these factors are at cross-purposes and require weighing one factor more heavily than another.

Visual communication, including symbols and icons, has three distinct interrelated dimensions: semantic, syntactic, and pragmatic. The strengths of icon design can be evaluated in terms of these basic components of communication. The semantic dimension refers to the relationship of a visual image to a meaning. Syntactic dimension refers to the relationship of one visual image to another, and pragmatic dimension refers to the relationship of a visual image to a user.

Icon images should be designed to meet unique communication needs, while maintaining a visual consistency throughout, using constant scale and limited size variations, orientation of figures with respect to text, use of colors, variation of line weights, and treatment of borders. These visual themes establish recognizability, clarity, and consistency while avoiding unnecessary variation of curves, line thickness, shape, color, and number of parts.

All icons, simple and complex, must function as a group with a recognizable visual vocabulary. Simplifying the images and amount of detail (e.g., eliminating unimportant features) results in

consistently bold and direct symbols. Using an optically consistent line weight creates unity. Softening the edges (e.g., with curves) establishes visual relationships throughout the group.

It is beneficial to incorporate testing procedures as integral parts of the symbol development process, and not solely as a post-design evaluation. Criteria other than understandability also require consideration for many applications. For example, in an operational setting, the ability to distinguish the icon during over-the-shoulder (supervisory) viewing should be considered.

### **7.3.1 Types of Icons**

Icons and symbols are most effective when they represent a service or concession that can be represented by an object and less effective when used to represent a process, activity, or complex interactions. Even an experienced user may become confused when trying to deal with large numbers of arbitrary icons. Four basic icon types are defined below and illustrated in Figure 7-2.

- **Resemblance** - Depict the underlying referent through an analogous image. The road sign is a good example.
- **Exemplar** - Serves as a typical example for a general class of objects. The knife and fork is used to represent restaurant services. This simple image is a very powerful depiction of the salient attributes associated with what one does in a restaurant.
- **Symbolic** - Convey the underlying referent at a higher level of abstraction than the image itself. The wine glass effectively depicts the abstract concept of "fragility."
- **Arbitrary** - Bears no relationship to the referent and therefore the association must be learned. The biohazard sign is a typical example of the arbitrary form where no physical or analogous correspondence exists between the symbol or icon and the intended meaning.

### **7.3.2 Icon Usage**

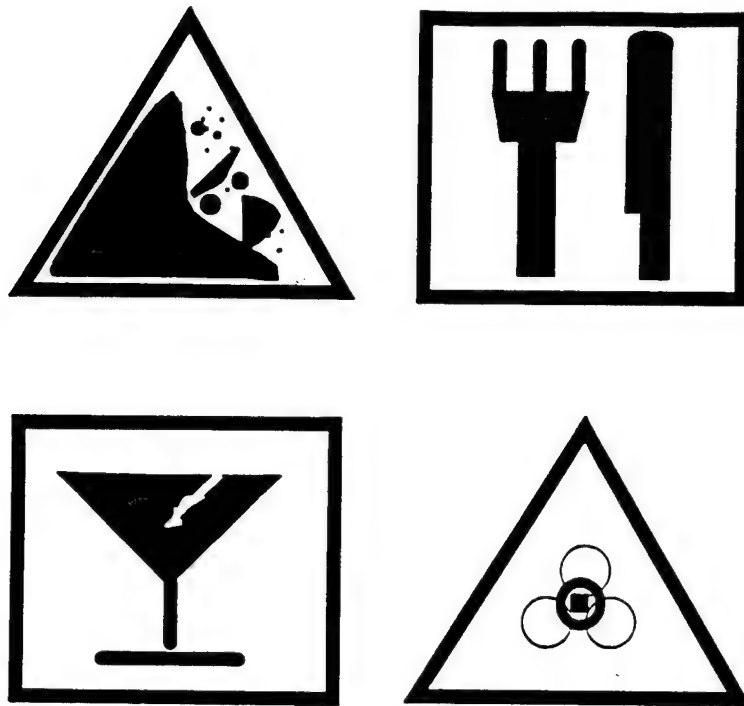
When using direct manipulation, use icons as visual representations of system functions available to the user. The larger the symbol set the more difficult it will be to learn the symbols.

#### **7.3.2.1 Iconic Menus**

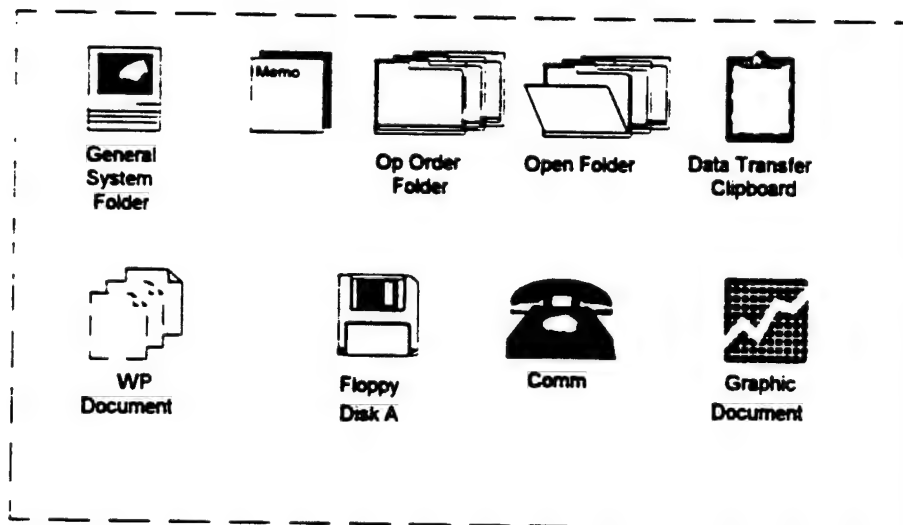
Iconic menus are groups of icons that act the same as textual menus, allowing selection of system options. Figure 7-3 illustrates an iconic menu.

- When users do not share a common language (e.g., NATO Forces), devise iconic menus for control functions. Test to be sure icon/text label is appropriate to many cultures. Examples from international signage may be appropriate.
- Place a limit on the number of icons shown at one time.





**Figure 7-2. Examples of Four Basic Icon Types**



**Figure 7-3. Example of an Iconic Menu**

- Divide the screen display into cells (grids) capable of holding one icon. Well ordered menu locations increase predictability and consistency as well as decrease clutter.
- Provide a consistent location for icons (i.e., frequently used icons should be positioned in the edges), but allow the user to customize locations during use.
- Ensure icon sizing and location are consistent with other aspects of the design (e.g., windows).
- Use existing icons when available.
- When using iconic menus, design the system such that once an action has been initiated through an icon (e.g., printing), nonselectable icons cannot be manipulated. Provide the user with a visual indication of which icons are unavailable (e.g., dimmed/shadowed appearance when unavailable).
- Highlight the icon when it is selected.

#### **7.3.2.2 Command Icons**

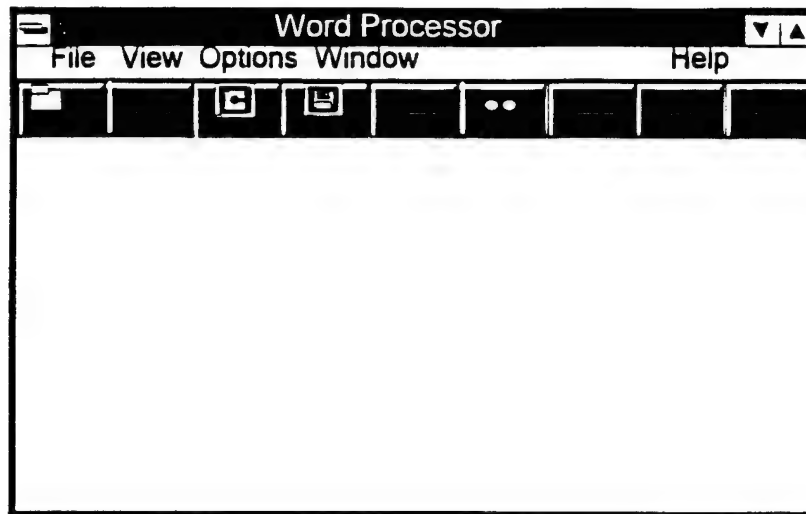
Command icons are computer icons representing frequently used computer commands and operations. Apply general design principles for icons to command icons.

- To the extent possible, ensure that command icons are standardized and consistent across all DoD applications (e.g., common set of icons for command and utility functions within tactical/operational applications).
- The greater the risk or danger, the more standardized the icon should be.
- Ensure that command icon meaning/function is consistent across displays and standardized within an application.
- COTS software must meet consistency requirements for icon use and design within an application (e.g., when COTS applications use metaphor/icon designs inconsistent the with suggested universal icon approach, ensure consistency within COTS software itself).

#### **7.3.2.3 Button Layout**

Horizontal or vertical layout of buttons is a new use of icons in COTS such as word processing packages. This feature provides quick access to frequently used commands and macros. The buttons on the bar perform the commands directly when selected with a pointing device. See example in Figure 7-4.

- Button layouts should be customizable. The user should be able to select functions and macros for inclusion on the button layout.



**Figure 7-4. Example of Icons Used on Button Layout**

- Design should allow selection with keystrokes as well as pointing device.
- Provide the capability to display and hide button layouts.

#### **7.3.2.4 Icon Mapping**

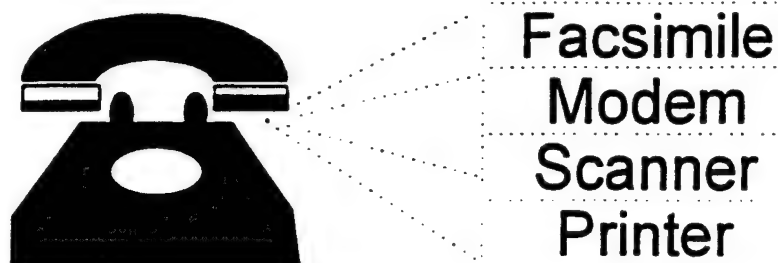
Iconic mapping implies the extent to which the link between the functionality depicted in the icon and the underlying functionality can be inferred. When an icon represents a group of functions (i.e., one to many mapping), do not repeat the specific icon for each function. Selecting the icon should cause an interface to appear that allows the user to select the specific function to be performed. For example, an icon for selecting communication devices, when selected, would bring up a window of types of communication devices available. The user would then select the appropriate device (see Figure 7-5).

#### **7.3.2.5 Switching to Textual Representation**

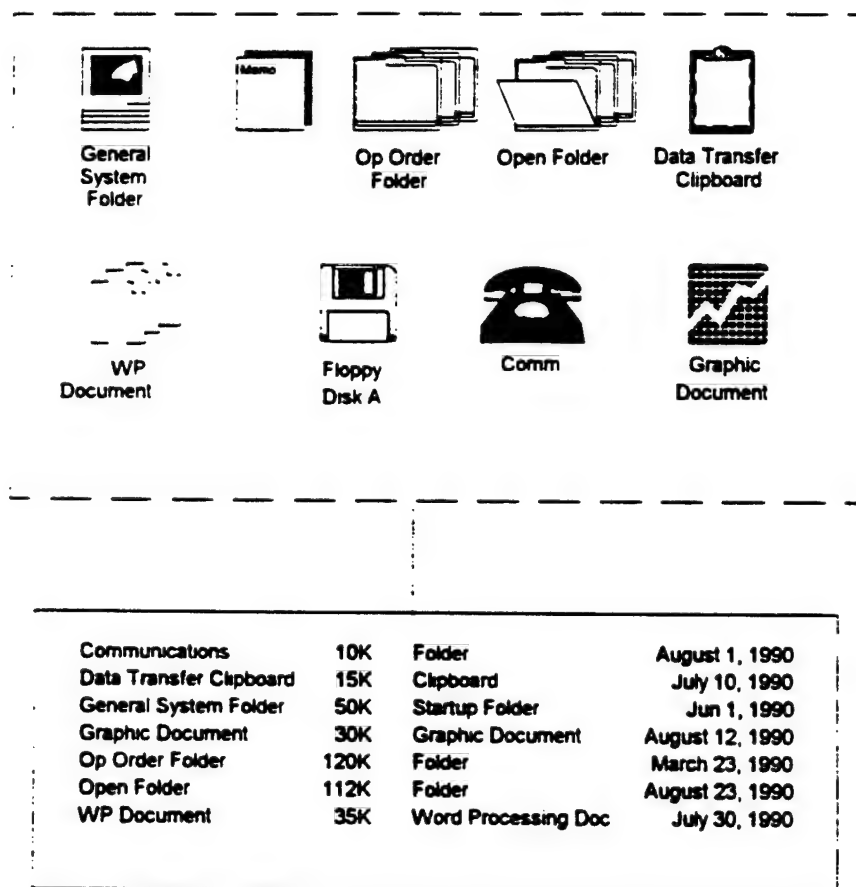
Include a feature that allows the user, when working with icons, to switch to a textual representation of the functions or files. The text should be listed sequentially to produce a logical transition from icon to text (see Figure 7-6).

### **7.3.3 Icon Design**

Basic principles for icon design are similar to those for designing large-scale windows and screens. In one survey, participants were asked to rank icons from most to least appropriate. The results are outlined below:



**Figure 7-5. Example of a Multi-Function Icon with Interface for Selection of Available Functions**



**Figure 7-6. Example of Textual Representation**

- Respondents preferred more concrete icons because it was easier to make association with something familiar. European respondents preferred more abstract icons, presumably because, in many everyday environments, they were more familiar with pictographic representations.

- Criticisms included that some icons were too similar and too numerous. Perceived similarity is a design concern because the ability to discriminate icons is an important feature.
- Some respondents asked for a box around all icons, while others disagreed. However, a box makes visual discrimination more difficult, especially for icons with image content near the perimeter of the box.
- Respondents requested a clearer indication of which activities (types or metaphors) should be represented by icons.

The designer should recognize that the ways in which a human perceives figures affects how icons are designed. The icon designer needs to incorporate Gestalt principles of human perception, briefly described as follows:

- Humans see the simplest or most efficient interpretation of an icon.
- The user will associate a meaning with an icon.
- Users tend to mentally group objects.
- Figure-ground relationships are important to how a user perceives an icon.

#### **7.3.3.1 Consistent Icon Design**

- Icon meaning should be consistent across displays and standardized within an application. To the extent possible, it should also be standardized and consistent across all DoD applications.
- As feasible, use a common set of primitives (software code that defines a specific shape, form, or color) and boundaries for icons. This will improve the user's ability to recognize and associate icons with their meanings.
- Ensure that icon is distinguishable from other icons (e.g., it shouldn't be similar to or confused with others).
- Ensure the icon can be seen well from all angles.
- In general, read icon pictures as books in the western culture, from top to bottom, left to right.
- Users prefer concrete symbols over abstract ones and simplicity over complexity.
- Design must avoid ambiguity. Ensure that no more than one meaning can be attributed to an icon.
- Use care when transferring design principles from one environment to another.

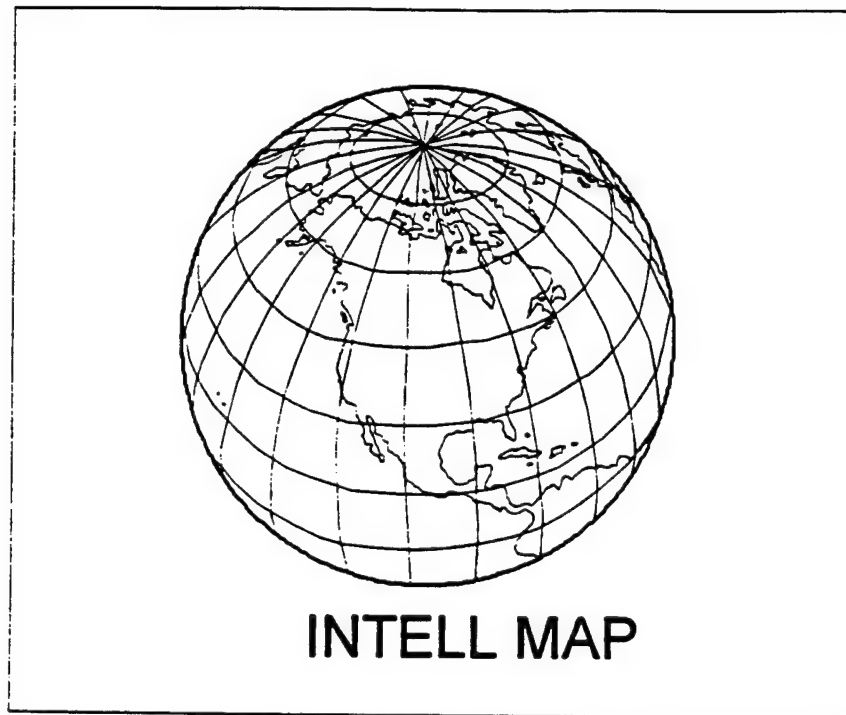
- Ensure that the user cannot select an invalid icon choice. Make unselectable those icons that should not be selected, and provide visual indication of this to the user.
- As a minimum, ensure that every icon includes the attributes of color, location, and visibility.
- Use existing icons that the user will recognize.
- Where possible, do not use icons unique to an application.
- Avoid the use of company logos and other such unique icons.

#### **7.3.3.2 Use of Color**

- Research shows that color offers no special advantage in speed of recognition. It is recommended that icon design be based on black and white rather than on color.
- Use caution when color-coding, and use color only if it is redundant to another coding method. See Subsection 4.3 for additional color guidelines.
- Use color with discretion. Too much color variation will confuse the viewer by creating clutter.
- In general, for color display, use five or fewer colors including black, white, and/or gray for icons.
- Use simple color patterns for background or low light areas.
- Limit colors to a carefully chosen set, and use them consistently across content areas and different display media.

#### **7.3.3.3 Icon Labeling**

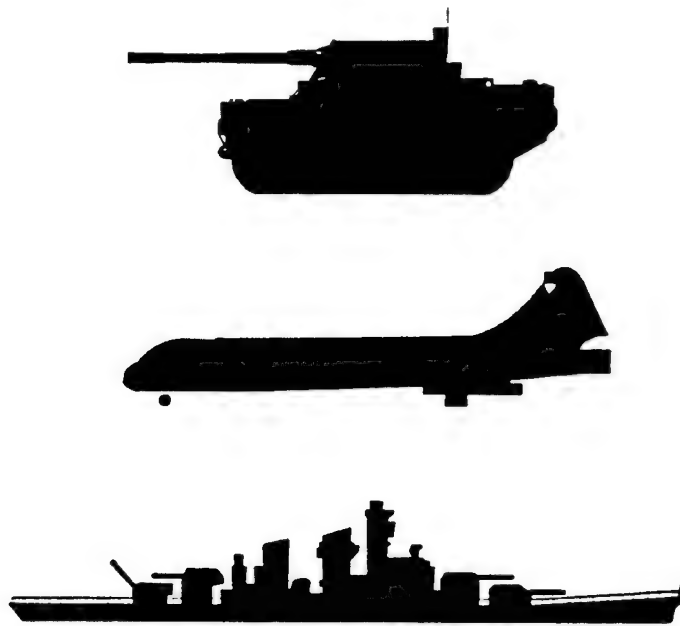
- An icon shape generally represents a class of items or functions. To distinguish the precise function, provide a text label that names each icon. Pictures are remembered better if they have been named.
- Place Icon labels underneath the icon, as illustrated in Figure 7-7. If labels are not used, the user should be able to query the system to get a definition of the icon.
- Keep textual material simple in icon labels.



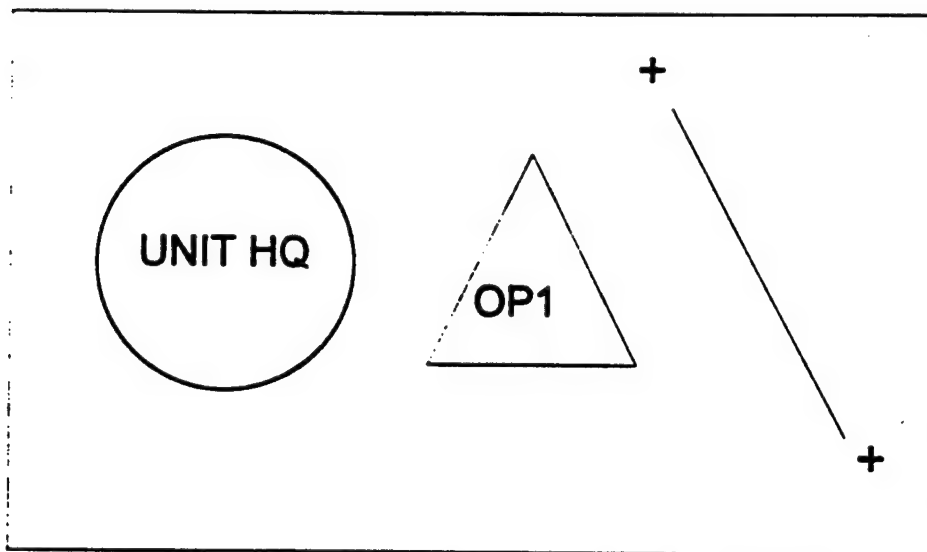
**Figure 7-7. Example of a Text Title for an Icon**

#### **7.3.3.4 Icon Shape**

- Ensure that icon shapes provide a visual representation matching user expectations and allow association between the icon and function being controlled.
- Design Icon shape as a concrete, not abstract, concept with respect to user.
- Ensure that icon shapes are as simple as possible to ensure user recognition. If icon shape is too complex, the user may make errors in recognizing the icon. Icon shape should show or exaggerate an object's natural features (see Figure 7-8).
- The fewer unique icon shapes used, the more effective the user-computer interface. The most basic (icon) library should be composed of rectangles, triangles, circles, arcs, lines, splines, text, and bitmap images. At a maximum, no more than 20 unique shapes should be used.
- Icons to be used with different cultural or national groups (e.g., NATO Forces), should use technological shapes or forms rather than natural objects. The examples shown in Figure 7-9 include areas shaped as circles, triangles, or boundaries shaped as lines with plus signs (+) as end marks.



**Figure 7-8. Examples of Icon Shapes**



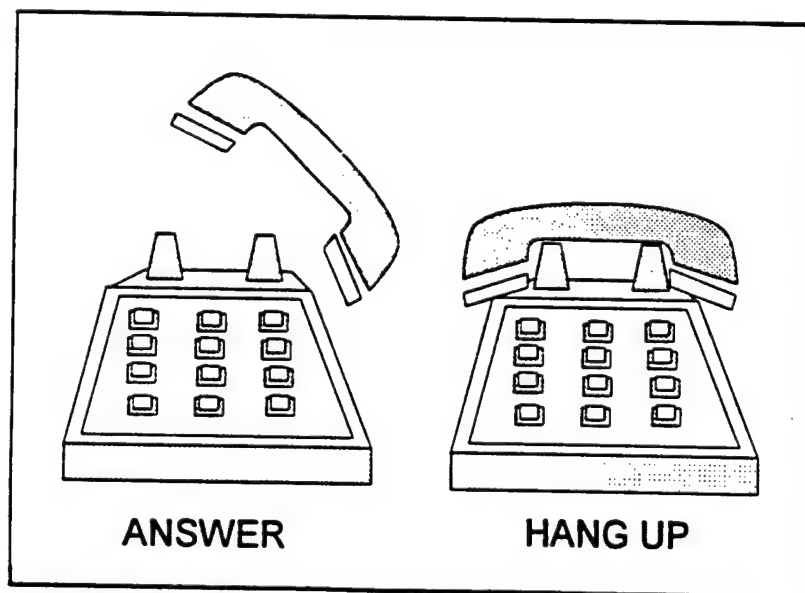
**Figure 7-9. Example of the Use of Technological Shapes**



- Icons should be consistent with international usage - triangle is “caution” or any pointed object is “enemy,” circle with diagonal line through is “prohibit,” square is “potential danger or unknown,” and round or curved objects are “friendly.”
- When icons represent opposite functions, design the icons so they mirror one another (see Figure 7-10).
- Orient figures consistently with respect to text.
- If 3D icons are used, allow the user to manipulate them through rotation so they can be viewed from different vertical and horizontal viewing angles.

#### 7.3.3.5 Icon Size

- Icons should be at least 1/4 inch in height on the screen to reduce the time required for positioning the pointer on the target and performing the required controlling actions.
- When size coding of icons is used by operational systems, the larger icon should be 1.5 times as large as the next smallest.
- Up to five sizes of icon can be used for coding, but no more than three are recommended for operational systems.
- Icons should have a size ratio to the background of 1:1.5 for best visual discrimination.
- Do not use the same symbol in different sizes to mean different things.
- Scales should be kept constant.



**Figure 7-10. Example of Mirrored Icons**

### **7.3.3.6 Grouping Icons**

- Base the grouping of icons on the proximity, similarity, and arrangement of objects that define a closed region. Arrange objects in straight or smoothly curving lines. Ensure that symmetrical icon arrangement is maintained when icons undergo simultaneous, correlated changes.
- Grouping can provide additional meaning to icons, but the designer should ensure that unrelated icons are not inadvertently grouped by users.
- Visually separate images in 2-dimensional space.
- Place a finite limit on the number of icons shown at one time.
- Divide the screen into "cells" to improve location and relocation of icons. A well ordered location increases predictability and consistency as well as decreases clutter.

### **7.3.3.7 Figure-Ground Relationships**

Figure refers to an object, which has a shape and stands out from the background. Ground refers to the area that is perceived to continue behind the figure. Keep in mind the following points when designing icons:

- The size of a figure relative to its background is important. The smaller the size of a figure relative to the background, the more likely it will be perceived as a figure.
- When shape only is used as a discriminator for figures, convex shapes will likely be perceived as figures and concave shapes as holes.
- A contour line will be perceived as belonging to only one of the areas it delineates.
- Position affects whether an object will be perceived as a figure. Centrally positioned objects and the lower portion of a surface divided horizontally into two parts are seen as figures.
- The greater the contrast between an object and its background, the greater the perception of the object as a figure.
- Icon figure-ground (foreground lines, etc.) should be clear and stable.

### **7.3.3.8 Icon Boundary Lines**

- Icon boundary lines should be solid and closed and should have a high contrast value (e.g., the best boundary is based on the contrast between the figure and the underlying display background).
- Use an optically consistent line weight for unity; use curves to enhance visual relationships.

- Corners should be smooth.
- It is best not to put a box around an icon. The box makes visual discrimination more difficult.
- If a boundary is left open, be aware that the user will tend perceptually to close the open boundary. Design the opening to ensure only desired closures will occur.

#### 7.3.3.9 Meaning

The user associates a meaning with an icon. Icons can be a very powerful form of communication since they have the potential of being universally meaningful.

- The meaning should be inherently obvious. The stronger the associated meaning, the more easily the icon will be recognized and remembered.
- Icons should have intrinsic meaning to the user. Care must be taken to ensure no negative connotations can be attributed to the icon.
- Avoid icon designs that could have a range of attributed meanings. The user should not have to look up a meaning or activate an icon to understand what it does or means.
- Carefully consider icon style. Representational and abstract symbols have visual resemblance; arbitrary or invented symbols must be learned.
- Learned icons should be as unique or compact as possible to aid in training the user.
- Icons should not be too realistic, stylized, simple, or complex.
- Design icons as concretely as possible. Subjects tend to rate less concrete icons as more appropriate when they have been redesigned to be more concrete. Appropriateness has been found to be a reasonable predictor of icon identification time.
- Grouping should be used to provide additional meaning to icons.

#### 7.3.3.10 Hardware Considerations

- Computer monitors should offer sufficient resolution so that the icons can be identified by the user at normal viewing distance.
- On high-resolution screens, at 60-150 dots per inch (dpi), 30-60 pixels per inch are often used.
- The designer should keep in mind that display standards, such as color graphics adapter (CGA), enhanced graphics adapter (EGA), and video graphics array (VGA) screen resolutions, are all different; some even have differently shaped pixels. Icons must be designed to appear correctly in each of these screen display standards.

- If icon display equipment has severe limitations in appearance or interaction characteristics (e.g., monochrome CRTs or touch-screen input), this will affect the appearance of icons and their use by the viewer.
- The designer should consider the intended display medium when designing the icon (e.g., an icon design for a high-resolution display may lose meaning when presented on a low-resolution display).

#### **7.3.4 Design Methodology**

Successful icon design involves approaching the problem systematically by analyzing the sometimes conflicting needs that determine appearance and interaction characteristics, designing prototypes, and evaluating the design.

Although no set of rules can guarantee a perfectly designed icon or set of icons, the following general design steps are suggested.

##### **7.3.4.1 Analyze Contents**

Analyze the verbal contents and the display environment to determine how icon parts and complete icons should relate.

##### **7.3.4.2 Use Sketches**

Specify appearance of the icon, placing and shaping instances of existing icons. Design the initial icons by creating quick sketches showing all visual elements, their approximate size, and approximate location. It is easier to manipulate broad differences in icons and their hierarchy early in the design process, but avoid becoming too precise. Explore all possible design variations.

##### **7.3.4.3 Establish Style**

A consistent stylistic treatment has a perceived complexity of the icons. Styles should be established in which the icons are grouped by consistent approach or appearance. User involvement is a critical factor at this point.

##### **7.3.4.4 Establish Layout**

Consider the following issues for screen layout design:

- An underlying grid helps organize major elements of the icons to make all visual components consistent (e.g., point elements, gray patterns, curves, angles, length and width of rules).

- Establish standard horizontal, vertical, and oblique lines and a limited set of sizes for objects. Use the grid to regulate groups of text and images and determine the size of elements in order to build a visual consistency.
- Use an articulate, systematic method of assigning areas for text and illustration, as well as for background field or format. When possible, use strong, easily recognized proportional format.
- Research concerning the user's ability to select images on a CRT screen supports the interpretation that the location of icons is of high importance; frequently used icons should probably be positioned around/near the edges of the screen.
- It is important to make room for the most important icons in the same places on screen (e.g., the trash bin).

#### **7.3.4.5 Distinguish Icons**

Consider the following items to distinguish icons.

- Use large objects, bold lines, and simple areas.
- Select a style of presentation, and continue to use it within the icon set.
- Avoid sudden changes in emphasis or de-emphasis of certain objects, structures, or processes.
- Ensure crucial elements are of sufficient size in comparison to the total size of the icon.

#### **7.3.5 Icon Evaluation**

The following are some issues to use for evaluating symbols:

- Is symbol/message association easy? Representational and abstract symbols have visual resemblance; arbitrary or invented symbols must be learned (e.g., math symbols). Arbitrary symbols should be as unique or compact as possible, as they require training.
- In a variety of cultures and situations, is the symbol equally appropriate? Icons created for one cultural group may generate incongruent or even opposite meaning for another group. Such differences may generate conflicts in communications.
- Will the symbol be appropriate in the future? Will the metaphor soon be obsolete?
- Is the symbol pleasing and noncontroversial?
- Is the symbol in accordance with existing international standard symbols (i.e., do not create new symbols if one already exists)?

- Can the symbol or its elements be applied systematically for a variety of interrelated concepts (i.e., can the elements form a rich symbolic language, combining them to form more complex symbols)?
- Is the symbol easy to reproduce in a variety of environment and situations? Can it be transferred to different systems, enlarged or reduced without losing crispness or detail?
- Is the symbol distinguishable from other symbols?
- Can the symbol be perceived from different distances, angles, conditions?
- Do icons have intrinsic meaning to the user? The user will associate a meaning with an icon. The stronger the associated meaning, the more easily the icon will be recognized and remembered.
- Do icons provide a visual representation that matches user expectations and allows association between the icon and the function being controlled?

#### **7.3.5.1 Testing Icon Design**

It is imperative to test icon design with a group of users who represent the intended user. This should ensure that the meaning of the icon is implicitly understood.

#### **7.3.5.2 Usability**

- Icons should be general enough to allow the user to understand and use other metaphors or media, such as text-based systems.
- Application software should allow the creation of icons that represent macro instructions. The user will be able to use these macros more effectively with the advantages of a visual representation.

## REFERENCES

Paragraph	References
7.0	Helander (1988); Gittins (1986); Lewis and Fallesen (1989); DISA/CIM (1992c) pp. 7-1 to 7-3
7.1	Smith and Mosier (1986) para 3.1.8-5
7.1.1a	Ziegler and Fähnrich (1988) p. 127
7.1.1b	Ziegler and Fähnrich (1988) p. 127
7.1.1c	Ziegler and Fähnrich (1988) p. 128
7.1.2	Ziegler and Fähnrich (1988) p. 128
7.1.3	Ziegler and Fähnrich (1988) p. 129
7.1.4	Bowser (1991)
7.2	Lewis and Fallesen (1989) p. 91; Gittins (1986) p. 519; Lerner and Collins (1980) p. 32; Tzeng et al. (1990) pp. 77-97
7.2.1	Arnstein (1983) preface
7.2.1.1	OSF (1990) p. 531
7.2.1.2	Lewis and Fallesen (1989) p. 92
7.2.1.3	DISA/CIM (1992c) pp. 7-2
7.2.1.4	DISA/CIM (1992c) pp. 7-2
7.2.1.5	Lewis and Fallesen (1989) p. 92; Gittins (1986) p. 530 and 540
7.2.2.1	Gittins (1986) p. 528
7.2.2.2	OSF (1990) p. 531
7.2.2.3	Lewis and Fallesen (1989) p. 91
7.2.2.4	Lewis and Fallesen (1989) p. 91; Gittins (1986) p. 540
7.2.2.5	OSF (1990) p. 530
7.2.2.6	Lewis and Fallesen (1989) p. 91; Gittins (1986) p. 540
7.3	AIGA (1982) preface, p. 129; Marcus (1979) p. 26; Tzeng et al. (1990) p. 78; Lerner and Collins (1980) p. 32
7.3.1	AIGA (1982) p. 11; Wood and Wood (1987) p. 100; Rogers (1989) p. 110

## REFERENCES (cont'd)

Paragraph	References
7.3.2	Ziegler and Fähnrich (1988) p. 127; Jorna (1988) p. 182
7.3.2.1a	Smith and Mosier (1986) para 3.1.8-5; Wood and Wood (1987) p. 100
7.3.2.1b	Gittins (1986) p. 532
7.3.2.1c	Gittins (1986) p. 532
7.3.2.1d	Gittins (1986) p. 535; Blankenberger and Hahn (1991) p. 376
7.3.2.1e	Bullinger et al. (1987) p. 90; Gittins (1986) p. 536
7.3.2.1f	DISA/CIM (1992c)
7.3.2.1g	Gittins (1986) p. 526
7.3.2.1h	Lewis and Fallesen (1989)
7.3.2.2	DISA/CIM (1992c)
7.3.2.2a	DISA/CIM (1992c)
7.3.2.2b	Wood and Wood (1987) p. 103
7.3.2.2c	DISA/CIM (1992c)
7.3.2.2d	DISA/CIM (1992c)
7.3.2.3	WordPerfect Corporation (1991) pp. 35-42
7.3.2.4	Rogers (1989) p. 110
7.3.2.5	Ziegler and Fähnrich (1988) p. 128
7.3.3	Marcus (1992) p. 35; Nolan (1989) p. 381; Lewis and Fallesen (1989)
7.3.3.1a	Lewis and Fallesen (1989) p. 89; MacGregor, King, and Clarke (1988) p. 275
7.3.3.1b	Lewis and Fallesen (1989) p. 89
7.3.3.1c	Wood and Wood (1997) p. 101
7.3.3.1d	Wood and Wood (1997) p. 101
7.3.3.1e	Neurath (1980) p. 49
7.3.3.1f	Stammers and Hoffman (1991) pp. 354-356
7.3.3.1g	Rogers (1989) p. 106



## REFERENCES (cont'd)

Paragraph	References
7.3.3.1h	Stammers and Hoffman (1991) pp. 354-356
7.3.3.1i	Gittins (1986) p. 526
7.3.3.1j	Rosenstein and Weitzman (1990) p. 525
7.3.3.1k	DISA/CIM (1992c)
7.3.3.1l	DISA/CIM (1992c)
7.3.3.1m	Wood and Wood (1987) p. 103
7.3.3.2a	Tullis (1981) p. 548
7.3.3.2b	Lewis and Fallesen (1989) p. 91; Gittins (1986) p. 539
7.3.3.2c	Marcus (1992) p. 63
7.3.3.2d	Marcus (1992) p. 63
7.3.3.2e	Marcus (1992) p. 63
7.3.3.2f	Marcus (1984) p. 26
7.3.3.3a	Lewis and Fallesen (1989); Ziegler and Fähnrich (1988) p. 128; Smith and Magee (1980) p. 384
7.3.3.3b	Lewis and Fallesen (1989) p. 91
7.3.3.3c	Neurath (1980) p. 24
7.3.3.4a	Ziegler and Fähnrich (1988) p. 128; Smith and Mosier (1986) para 2.4-13; Shneiderman (1987) p. 200; Lewis and Fallesen (1989) p. 89; Lansdale (1988) p. 148
7.3.3.4b	DISA/CIM (1992c)
7.3.3.4c	Ziegler and Fähnrich (1988) p. 128; Shneiderman (1987) p. 200; Lewis and Fallesen (1989) p. 89; Gittins (1986) p. 528
7.3.3.4d	Lewis and Fallesen (1989) p. 91; Rosenstein and Weitzman (1990) p. 525
7.3.3.4e	Lewis and Fallesen (1989) p. 91; Gittins (1986) p. 539
7.3.3.4f	Wiebe et al. (1993) p. 55
7.3.3.4g	Lewis and Fallesen (1989) p. 90
7.3.3.4h	Marcus (1979) p. 28

## REFERENCES (cont'd)

Paragraph	References
7.3.3.4i	Lewis and Fallesen (1989) p. 92
7.3.3.5a	Lewis and Fallesen (1989) p. 90
7.3.3.5b	Lewis and Fallesen (1989) p. 90
7.3.3.5c	Lewis and Fallesen (1989) p. 90
7.3.3.5d	Lewis and Fallesen (1989) p. 90
7.3.3.5e	Neurath (1980) p. 47
7.3.3.5f	Marcus (1979) p. 26
7.3.3.6a	Lewis and Fallesen (1989)
7.3.3.6b	Lewis and Fallesen (1989)
7.3.3.6c	Snyder (1987) p. 147
7.3.3.6d	Gittins (1986) p. 532
7.3.3.6e	Gittins (1986) p. 532
7.3.3.7	Lewis and Fallesen (1989); Gittins (1986) p. 539
7.3.3.8a	Lewis and Fallesen (1989) p. 90; Gittins (1986) p. 539
7.3.3.8b	AIGA (1982) p. 129
7.3.3.8c	Lewis and Fallesen (1989) p. 90; Gittins (1986) p. 539
7.3.3.8d	Nolan (1989) p. 381
7.3.3.8e	Gittins (1986) p. 539
7.3.3.9	Lewis and Fallesen (1989); Rogers (1989) p. 106
7.3.3.9a	Lewis and Fallesen (1989)
7.3.3.9b	Lewis and Fallesen (1989); Lodding (1983) p. 19
7.3.3.9c	Rogers (1989) p. 106
7.3.3.9d	Wood and Wood (1987) p. 100
7.3.3.9e	Wood and Wood (1987) p. 100
7.3.3.9f	Lodding (1983) p. 14
7.3.3.9g	Stammers and Hoffman (1991) p. 354
7.3.3.9h	Lewis and Fallesen (1989)

## REFERENCES (cont'd)

Paragraph	References
7.3.3.10a	Marcus (1992) p. 55
7.3.3.10b	Marcus (1992) p. 55-56
7.3.3.10c	Marcus (1992) p. 56
7.3.3.10d	Marcus (1992) p. 61
7.3.3.10e	Ziegler and Fähnrich (1988) p. 127
7.3.4	Marcus (1992)
7.3.4.1	Marcus (1992)
7.3.4.2	Marcus (1992)
7.3.4.3	Marcus (1992)
7.3.4.4	Marcus (1992)
7.3.4.4a	Marcus (1992); Marcus (1984) p. 26
7.3.4.4b	Marcus (1992); Marcus (1984) p. 26
7.3.4.4c	Marcus (1992)
7.3.4.4d	Blankenberger and Hahn (1991) p. 367
7.3.4.4e	Blankenberger and Hahn (1991) p. 374
7.3.4.5	Marcus (1992)
7.3.4.5a	Marcus (1992)
7.3.4.5b	Marcus (1992); Jorna (1988) pp. 173-183
7.3.4.5c	Marcus (1992)
7.3.4.5d	Marcus (1992)
7.3.5	Marcus (1992); Kato (1972) p. 100
7.3.5a	Marcus (1992)
7.3.5.b	Marcus (1992)
7.3.5.c	Marcus (1992)
7.3.5.d	Marcus (1992)
7.3.5.e	Marcus (1992)
7.3.5.f	Marcus (1992)

## REFERENCES (cont'd)

Paragraph	References
7.3.5g	Marcus (1992)
7.3.5h	Marcus (1992)
7.3.5i	Marcus (1992)
7.3.5j	Marcus (1992); Lewis and Fallesen (1989)
7.3.5k	DISA/CIM (1992) p. 7-14
7.3.5.1	Smith and Mosier (1986) para 2.4-13; Lerner and Collins (1980) p. 32
7.3.5.2a	DISA/CIM (1992) p. 7-14
7.3.5.2b	DISA/CIM (1992) p. 7-14

## 8.0 COMMON FEATURES

This section describes features, functions, and field display formats that should be handled consistently by all DoD applications. Subsection 8.1 deals with issues that apply primarily to operational systems. Subsections 8.2, 8.3, and 8.4 apply to all DoD systems. Subsection 8.2 discusses the topic of HELP. Subsection 8.3 discusses those characteristics of interactive control that apply to all dialog types. Subsection 8.4 discusses function keys. Additional information on interactive dialog can be found in documents such as Smith and Mosier (1986), Helander (1988), or DoD (1989b).

### 8.1 TACTICAL SYSTEM COMMON FEATURES

This section describes features, functions, and field display formats that should be handled consistently by all DoD operational applications.

#### 8.1.1 Date/Time Display

When date and time information are displayed in digital form, the format should be as follows:

##### 8.1.1.1 Date

Use YYYYMMDD, where YYYY is the four digits of the year, MM is the month, and DD is the day (e.g., 19910104 specifies 4 January 1991), or DD MMM YYYY, where DD is the day, MMM is the month, and YYYY is the year (e.g., 04 JAN 1991). With the year 2000 approaching, it is widely recognized that storing only two digits to denote a year will cause serious system problems. The display and data fields of a date should comply with a four-digit entry for the year.

##### 8.1.1.2 Time

Use HHMM{SS}Z, where HH is the hour of a 24-hour day, MM is the minute, SS (optional) is the second, and Z is the time zone. Zulu (Z), or Greenwich Mean time in civilian terms, is the system standard and the default DoD display standard (e.g., 113024Z). Unless otherwise specified, use colons or spaces on the display or output format to make the format more readable (e.g., 11:30:24Z). To simplify data entry and avoid extraneous characters, generate colons or spaces as part of the form, and do not leave this to user discretion.

##### 8.1.1.3 Local Time

Allow users to specify local time on hard-copy output and soft-copy display, as desired (e.g., 11:30:24L). However, do not provide this option to users in operational systems where input and coordination are based on Zulu time.

#### **8.1.1.4 Date/Time Group**

Military services specify that Date/Time Group should be displayed as DDHHMMZ MMM YY, where DD is the day, HH is the hour of a 24-hour day, MM is the minute, Z is the time zone (defaults to Zulu), MMM is the month, and YY is the year (e.g., 041130Z JAN 91). This format for the display of a date in no way implies that the data field should use a two-digit field for the year. While displays of the year may be abbreviated, data fields need to comply with a four-digit entry for the year.

#### **8.1.2 Latitude/Longitude Display**

Latitude and longitude displays will always be presented as two fields. The labels may be given as Lat and Long. When displaying latitude and longitude, use appropriate symbols for degrees, minutes, and seconds as part of the display. The formats are shown in the following two sections.

##### **8.1.2.1 Latitude**

Use D{D}H, where D (one or two characters) is the degrees of latitude and H is the hemisphere (N for North, S for South), or DD(MM{SS})H, where DD is the degrees of latitude, MM is the minutes of latitude, SS is the seconds of latitude, and H is the hemisphere (N for North, S for South).

##### **8.1.2.2 Longitude**

Use D(D{D})H where D (one, two, or three characters) is the degrees of longitude and H is the hemisphere (E for East, W for West), or DDD(MM{SS})H where DDD is the degrees of longitude, MM is the minutes of longitude, SS is the seconds of longitude, and H is the hemisphere (E for East, W for West).

#### **8.1.3 User-Definable Parameters**

Enable all users to configure their computer screens to individual preferences. User-definable parameters include, but are not limited to, those that follow.

##### **8.1.3.1 Display Colors**

Where feasible, users should be able to select map and window background colors from a color palette within a user-parameter selection window. The selected color should be immediately reflected in a sample item displayed within the selection window. However, users should not be allowed to change security banner colors or colors with specific tactical coded meaning. Other restrictions are noted in the service-specific addenda to this *Style Guide*.

##### **8.1.3.2 Printer Default**

In networked environments, enable users to specify the printer destination.

### 8.1.3.3 Mouse Button Function Mappings

Enable users to specify either left-handed or right-handed button configurations as defined in Section 3.0 of the *Style Guide*.

### 8.1.3.4 HELP Level

Enable experienced users to bypass novice-level HELP messages that are beneficial to a new user.

### 8.1.4 Wild-Card Characters

Use wild-card characters in queries and searches to support patterns. The use of wild cards is application-specific. Some applications may disallow wild cards or restrict their use to only a few of the following wild card conventions. Use the following conventions where possible.

#### 8.1.4.1 Single Alphabetic Character

Use an @ to replace any single alphabetic character (a-z and A-Z). For example, an input of abc@d would match abcad, abcd, and abczd, but would not match abc7d or abcd4d.

#### 8.1.4.2 Single Numeric Character

Use a # to replace any single numeric character (0-9). For example, an input of 123#4 would match 12334, 12394, but would not match 123x4 or 123554.

#### 8.1.4.3 Single Alphanumeric Character

Use a ? to replace any single alphanumeric character (a-z, A-Z, 0-9, and punctuation marks). For example, an input of abc?d would match the character strings abcad, abc(d, abc'd, and abc7d, but would not match abcx4d.

#### 8.1.4.4 String

Use an \* to replace zero or more alphanumeric characters. For example, an input of abc\*d would match the character strings abcad, abcd, abckjfi(rjk)fid, and abc7d, but would not match abcd5.

## 8.2 ON-LINE HELP

On-line help (HELP) provides procedural aids, the ability to recover from errors, and advice without requiring the user to exit the application. Ideally, HELP is always available. A well designed system offers context-sensitive HELP.

Two elements are critical to HELP: user interface and content; both are equally important (Kearsley 1988). HELP must be easy to use and provide readily understandable user guidance;

HELP must not add problems or make a user situation more confusing. The HELP interface design will contribute to how often HELP is used, because the more difficult the interface is to use or access, the higher the probability HELP will not be used. No HELP application is useful if difficult to obtain, hard to use, or difficult to return to the application program.

Computer users want to accomplish a particular task quickly and with the least effort possible. When users encounter a problem, they want a solution that involves minimal interruption of the task at hand. If information is not immediately available, users often guess, repeat a previous sequence, or ignore what is not understood. These responses usually lead to further problems.

Along with individual differences, users have varying degrees of computer experience. The following three types of user group may require different types or levels of HELP:

- Novices (users who have little experience with computers) need help with basic concepts and operations. Novices usually want to see only necessary information.
- Experts (experienced computer users) want to know about limitations, shortcuts, complex operations, and anything else that will allow them to do their work more efficiently.
- Casual users (who may be either novices or experts) only occasionally use a computer or current application. They may need help remembering aspects of the application they previously learned.

General guidelines:

- Make HELP easy for users to access.
- Make HELP available throughout the application.
- Make access to HELP uniform.
- Make HELP easy to understand.
- Make it easy to return to the application.

### **8.2.1 Types Of HELP**

HELP should reflect user requirements with no significant impact on application response time. Of the following three types of HELP, the advice and active forms are preferred. Embedded training is often included in HELP. However, it is recommended that embedded training not be combined with HELP.

#### **8.2.1.1 Advice**

Enable users to obtain advice from HELP. As users query HELP, they find an interactive, context-sensitive source of information that indicates the entry to make at the current location in the application, the required keystroke, or the steps to take to complete the task.



### **8.2.1.2 Active**

Ensure that HELP is active, such that when HELP application software senses an inappropriate entry, it interrupts to ask users what they are attempting and if they are sure they want to complete the operation they initiated. HELP then suggests the correct form or keystroke.

### **8.2.1.3 Passive**

Enable users to query HELP when they need assistance. The information may be in the form of on-line system documentation, such as a user's guide or a list of functions performed by combinations of keypresses.

## **8.2.2 General Design**

### **8.2.2.1 Minimize Keystrokes**

Provide single keystroke access to and exit from HELP.

### **8.2.2.2 Provide Memory Aids**

Assume users cannot remember everything required to run the application; provide memory aids.

### **8.2.2.3 Include Basic Information**

Include basic information you would expect only novices to seek.

### **8.2.2.4 Expand Upon the Manual**

Provide clearer explanations of information in the manual, using subsequent screens as needed. Do not simply repeat phrases from the manual that the user has read but may not understand.

### **8.2.2.5 Choose On-line Portions of the Manual Selectively**

Be selective when putting information on-line from the user manual. Do not put the entire manual on-line as this would make it more difficult to navigate and read through than the hard-copy version. It would also waste system memory.

### **8.2.2.6 Include Obvious Information**

Include all pertinent information, even that which may appear obvious to the developer.

### **8.2.2.7 Avoid Jargon**

Avoid using jargon. A friendly, effective interface is the most important component of a HELP system. It frustrates a naive computer user to type "HELP," then receive a bit of cryptic jargon

in reply. Use jargon common to all users, not of the designer or programmer, when use of jargon is unavoidable.

#### **8.2.2.8 Do Not Overload the User**

Do not expect the user to read more than three HELP displays at a time or to remember more than about five points.

#### **8.2.2.9 Do Not Use HELP to Teach**

Do not use HELP to teach novices how to operate the system. Provide step-by-step instructions to remind occasional users how to perform the most common tasks. Remember that most users perform the same few tasks over and over, in the simplest possible way.

### **8.2.3 Accessibility Of HELP**

#### **8.2.3.1 Universal Access**

Provide access to HELP from every screen.

#### **8.2.3.2 Availability**

Remind users that HELP is easily available by displaying the command or function key used to get HELP.

#### **8.2.3.3 Display HELP Status**

Display a message indicating the status of HELP availability, if HELP is not available at all times or places in the program.

#### **8.2.3.4 Single Action to Invoke**

Enable users to get HELP using only a single keypress or mouse-click.

### **8.2.4 Provide HELP on HELP**

#### **8.2.4.1 Alphabetical Index of Functions**

Make an alphabetical index of HELP functions available to the user.

#### **8.2.4.2 Alphabetical Index of Commands**

Provide an alphabetical index with explanations of all commands used by the application software, showing the argument options.

#### **8.2.4.3 Show How to Use**

Show users how to use the HELP function. Never assume that HELP is obvious, even to expert users.

#### **8.2.4.4 Present Alternatives**

Show how to get HELP from anywhere in the system. Because users may know only one route, detail alternatives, including how quick and easy it is to use the options. Define different meanings of the HELP display, and explain their functions.

#### **8.2.4.5 Navigating Through HELP**

Show how to navigate within HELP. Explain how to scroll or page through a topic and how to jump to related topics.

#### **8.2.4.6 Provide HELP on Screens and Windows**

Describe the current window, including its function and tasks the user can perform.

#### **8.2.4.7 Provide Instructions to Novices**

Place instructions for using HELP on every HELP display, to assist novice and casual users.

#### **8.2.4.8 Instruct on When to Use**

Provide users with complete instructions on when to use the information supplied by HELP.

### **8.2.5 Application Information**

Provide a list of application capabilities. Show application components, options, and structure to help the user understand the application and use it more effectively. Experienced users as well as novices underutilize many applications because they do not recognize the full range of capabilities.

#### **8.2.5.1 Provide Shortcuts**

Use HELP to point out shortcuts and unused features.

#### **8.2.5.2 HELP on Messages**

Make available successively more detailed explanations of a displayed error message. HELP should be considered to provide more detailed messages, such as information and status.

### **8.2.5.3 HELP on Prompts and Definitions**

Make available successively more detailed explanations of a displayed question or prompt and definitions of specified terms.

### **8.2.5.4 Show Correct Input**

Provide examples of correct input or valid commands.

### **8.2.5.5 Show Command Format**

Provide a description of the format of a specified command and a list of allowable commands.

### **8.2.5.6 Provide User-Centered HELP**

Ensure that HELP is user-centered; base HELP on the user's task, not on application characteristics. Descriptions of application characteristics are more appropriate for a hard-copy user's manual.

## **8.2.6 Provide HELP In Context**

Context-sensitive HELP may be the most important kind of HELP for users. Ensure that context-sensitive HELP describes the nature of a specific control (check button, radio button, slider bar) and how people use that control.

### **8.2.6.1 Provide Specific HELP**

Ensure the HELP is specific to each level of user interaction (e.g., for context-sensitive HELP in a field specifying printer baud rate).

*Note: "Baud rate is the speed in bits per second at which your printer can accept data. Acceptable speeds are 1200, 2400, and 9600. Enter the speed in bits per second."*

### **8.2.6.2 Show Correct Alternatives**

List correct alternatives if the user enters an incorrect command.

### **8.2.6.3 Provide HELP Within Application**

Provide HELP within the application so users do not have to abandon their place in the application to seek HELP. Ensure that users do not have to close files, exit the application, and/or log off to invoke a HELP utility.

#### **8.2.6.4 Use Split Screen or Window**

Allow users to see the application screen that relates to the HELP request by means of a split screen or window. A separate HELP screen that completely replaces the application screen is undesirable because it prevents the user from simultaneously observing the problem and the HELP screen.

#### **8.2.6.5 Resize and Reposition HELP Windows**

Provide the user the capability to resize and reposition windows to see the HELP information and the problem at the same time.

#### **8.2.6.6 Identify Special Keys**

Display the meanings assigned by the application where applications have special uses for keys, especially function keys.

### **8.2.7 User Control of the HELP System**

Give users more control over a HELP system, as they will find this more useful.

#### **8.2.7.1 User-Initiated**

Allow users to initiate a HELP request and select the desired HELP topic.

#### **8.2.7.2 User-Selected Levels**

Allow users to select a level of HELP if multiple levels are available.

#### **8.2.7.3 Annotate Messages**

Allow users to annotate existing HELP messages.

#### **8.2.7.4 Describe Key Functions**

Provide the capability within HELP of pressing any key to obtain a list of features whose names begin with that letter. When the user selects a feature from the list by highlighting or clicking, provide an explanation of the feature.

### **8.2.8 Provide Consistent HELP Format**

#### **8.2.8.1 Consistent Screens**

Provide consistent HELP aids from screen to screen, both with indicators that HELP is available (e.g., "F1 = HELP") and the specific location on the screen.

### **8.2.8.2 Progressive Detail**

When providing progressively more detailed explanations, ensure the process of moving from level to level is consistent from screen to screen.

### **8.2.9 Self-Explanatory and Concise Displays**

#### **8.2.9.1 Match Titles to Contents**

Reflect or match the content of a HELP window in its title (e.g., the title of a HELP window for the entry field "Trans" could be "Help for Trans").

#### **8.2.9.2 Match Names**

Ensure that the name on the HELP display matches the panel from which help was requested (e.g., when working on an accident report, the help display may read, "HELP: ACCIDENT REPORT").

#### **8.2.9.3 Ensure Relevancy to the User**

Tailor the display to the current information requirements of the user, so only relevant data are displayed.

#### **8.2.9.4 Provide Clear Messages**

Make error and HELP messages clear, concise, and appropriate to the experience and training users have had in using the system.

#### **8.2.9.5 Use Task-Oriented Wording**

Adopt task-oriented wording for labels, prompts, and user guidance messages, incorporating whatever special terms and technical jargon may be normally employed in the user's tasks.

#### **8.2.9.6 Increase Understandability**

To increase understandability of HELP, apply the following principles:

- Use short sentences when writing HELP messages.
- Use the active voice in all HELP messages.
- Provide as many examples as possible for each HELP screen.
- Place HELP information in tables, where applicable.
- Put the answer before the explanation when presenting HELP information.

- Answer the most likely HELP questions immediately.
- Minimize the user requirement to scroll or page through displays.

## **8.2.10 Make Return To Application Easy**

### **8.2.10.1 Single Keystroke**

Enable the user to return to the application with only a single keypress or mouse-click.

### **8.2.10.2 Exit HELP Easily**

When a single keystroke exit is not possible, enable the user to return to the application easily. Ideally, this would be accomplished without calling up a menu and then choosing an item from it.

## **8.2.11 Keep HELP Current**

### **8.2.11.1 Provide Up-to-Date HELP**

Plan and build HELP concurrently with developing applications, so HELP information reflects the current version of the software. Provide updates to HELP with subsequent software releases.

### **8.2.11.2 Tailor HELP to the User**

Collect data on user target population to tailor HELP to the training and experience of the users.

## **8.2.12 Provide User Options**

### **8.2.12.1 Bookmarking**

Provide "bookmarking" so users can flag specific HELP messages for easy referral later. This can be especially useful in a large help system consisting of many topics and screens. Bookmarking allows users to customize the HELP system to their own needs and filter out information of no interest; it can speed up the HELP process and return the user to work with fewer interruptions.

- Allow the user to select a bookmark option while viewing the message to flag a HELP message.
- Ensure that user has an option to see all or just the bookmarked messages.
- Ensure a print option is available while HELP messages are being displayed. Users often want to print out HELP information to study it further.

### **8.2.13 System-Initiated Messages**

Provide system-initiated messages when an error has been detected or when other evidence reveals that the user is having a problem (e.g., missing parameters, duplicating erroneous commands, long lapses in response, out-of-range responses).

#### **8.2.13.1 Positive Tone**

Ensure that messages have a positive tone, indicating what must be corrected. Focus on correction(s) to the problem, not the action that caused it.

#### **8.2.13.2 User Control**

Present system-initiated messages to users as advice or suggestions. Ensure that HELP messages are not intrusive.

#### **8.2.13.3 Error Control**

Provide system-initiated HELP messages for systems where incorrect user actions could result in serious consequences. This is especially true for destructive actions such as deletions (e.g., MS-DOS command: "Del \*.\*"), file replacements, exiting an application without saving data, or renaming a file.

#### **8.2.13.4 Document Errors**

In error messages, always state the error detected, the input field containing the error, and the corrective action.

#### **8.2.13.5 User Understanding of Message**

Indicate clearly in messages whether they are meant to inform of error, indicate status, prompt for action, or provide feedback.

#### **8.2.13.6 User Options**

Give users the option to turn off system-generated messages or to specify the level or type of message to be given (e.g., advisory, caution, warning).

#### **8.2.13.7 Avoid Jargon**

Ensure that error messages are specific and address the problem in user terms. Avoid vague terms, such as "syntax error," or obscure error code numbers.



## 8.3 INTERACTIVE CONTROL

Interaction between the computer and the user is performed through a two-way communication process: 1) the user inputs commands, and 2) the computer responds to the input. Generally, two interchangeable names are given to this process -- sequence control (Smith and Mosier 1986) and interactive control (DoD 1989a). The term "interactive control" will be used in this *Style Guide*.

- Interactive control of a system occurs through a give-and-take of command and response between the user and the computer, called a "dialog." The following have been identified as the eight major types of user-computer dialogs (Smith and Mosier 1986):
  - **Question and Answer** - The user responds to questions posed by the computer.
  - **Form Filling** - The user enters a series of commands or data items in predefined fields. These fields may be mandatory or optional.
  - **Menu Selection** - The user selects from predefined option lists by pointing with a device, such as a mouse, or keying in associated codes.
  - **Function Keys** - The user controls the dialog by using fixed or variable function keys on the keyboard.
  - **Command Language** - The user makes control entries by composing specified messages for the computer.
  - **Query Language** - The user employs a specialized type of command language to elicit information from a computer system. This is used extensively with databases.
  - **Natural Language** - The user can compose messages to control the computer based on natural, not specialized, languages.
  - **Graphical Interaction** - The user makes selections and controls the computer interaction by direct manipulation.

Each of these eight types of dialogs can be used individually or combined as a suite or set of techniques. OSF/Motif, for example, supports a combination of menu and graphical interaction techniques.

Eight principles form the basis for designing a good human-computer dialog (Shneiderman 1987; Bailey 1982). These principles are as follows:

- Strive for consistency of design across terminology, menus, command structure, etc. for all applications.
- Enable frequent users to use shortcuts, improving user acceptance and overall system performance.
- Offer informative feedback for all user actions.

- Design dialogs to yield closure. The user will then feel a sense of accomplishment and will know when to go on to the next task.
- Offer simple error-handling, both by system error-checking and ease in correcting an identified error.
- Allow easy reversal of actions, such as an UNDO capability.
- Enable the user to feel in control of the interaction with the system.
- Reduce short-term memory load on the user by using intuitive displays, interactive sequences, sufficient training, and on-line helps and tutorials.

The primary dialog types used by applications share certain design considerations and guidelines. These are addressed in this section and organized into six topics: general, context definition, transaction selection, interrupts, error management, and alarms.

### **8.3.1 General**

The following general guidelines for interactive control apply to DoD systems.

#### **8.3.1.1 Displayed Context**

If the results of a control entry vary depending on a prior action of the user or computer, display a continuous indication of the current context (mode).

#### **8.3.1.2 Irrelevant Data**

Provide the user with the capability to remove irrelevant items from the display and to reverse this action (i.e., retrieve information that was removed).

#### **8.3.1.3 Page-Back Capability**

When the requested data exceed the capacity of a single display frame, provide the user with easy methods to move back and forth over displayed material by paging or panning/scrolling.

#### **8.3.1.4 Upper and Lower Case Equivalent**

For interpreting user-composed control entries, treat upper and lower case letters as equivalent.

#### **8.3.1.5 User-Callable Unfamiliar Term Descriptions**

- Write interface dialog to provide the capability for the user to call up descriptions of unfamiliar terms and commands through context-sensitive HELP screens. See Subsection 8.2 for additional information.

- Ensure that the interface displays to the user, as needed and in immediately usable form, terminology and commands necessary to perform the task associated with the displayed information.

#### **8.3.1.6 User-Paced Sequence Control**

Allow users to pace control entries, rather than forcing them to keep pace with computer processing or external events.

#### **8.3.1.7 Logical Transaction Sequences**

Design the sequence of transactions (e.g., number and sequence of steps in a task) from the perspective of what is logical to the user, not what is logical from the perspective of computer processing or ease of programming.

#### **8.3.1.8 Automated Information Entry**

Routinely and automatically include informational elements required for every communication or transaction after first input (e.g., call signs and authentication procedures).

#### **8.3.1.9 Customized Display/Control Options**

Allow the user to customize the information displayed on a screen to the particular tactical mission or scenario. For example, the user should have the flexibility to define which files can be displayed concurrently and what tactical data will be utilized in a single display. See Figure 8-1 for examples.

#### **8.3.1.10 Distinctive Display of Control Information**

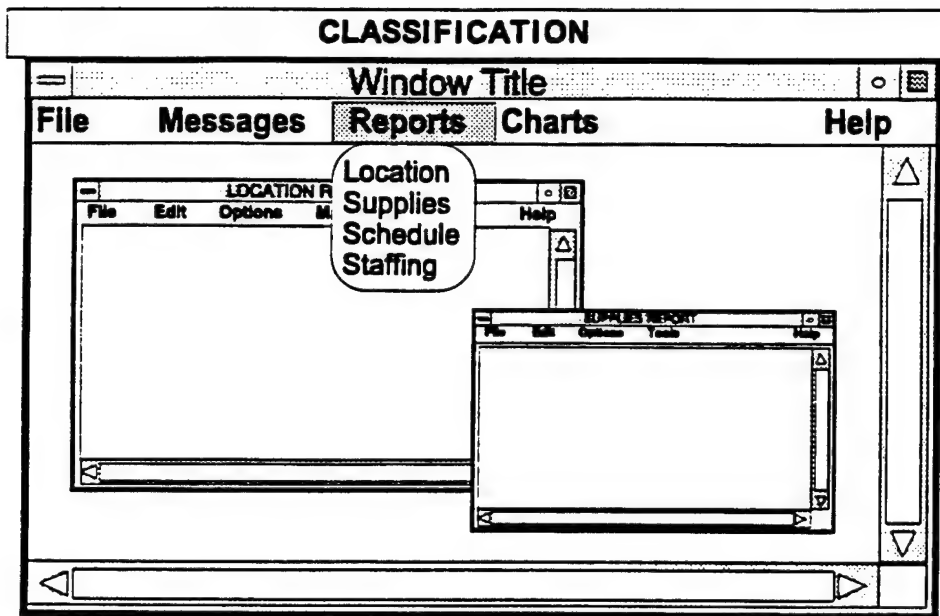
Design all displays so features (e.g., prompts, advisories, etc.) relevant to interactive control are distinctive in position and/or format.

#### **8.3.1.11 System Matched to User Abilities**

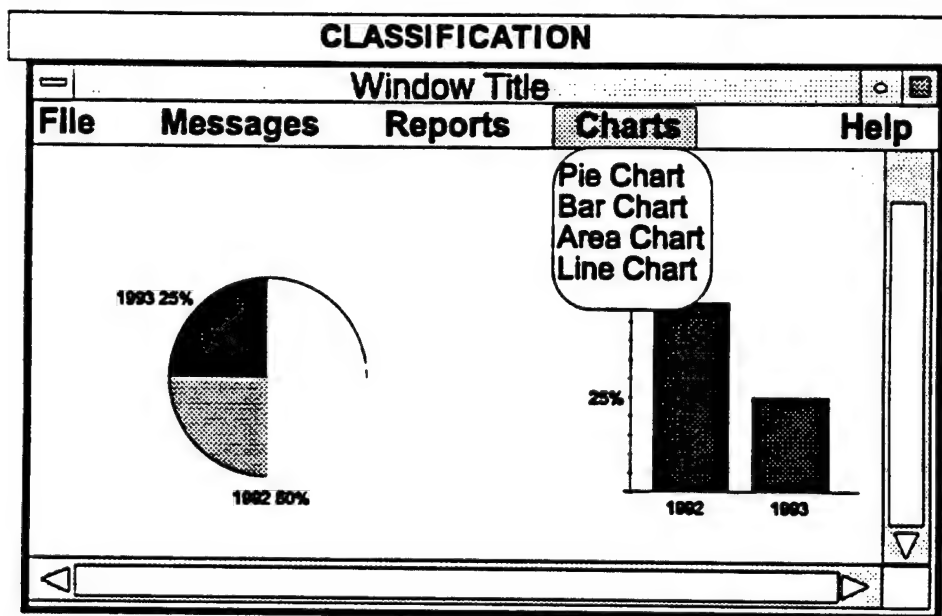
Ensure that applications adapt to individual differences and accommodate the variety of user abilities, whether novice or expert. For example, make accelerator keys for menu selection or command stacking available to the expert.

#### **8.3.1.12 Response Demand**

Demand response from the user while instructions on how to respond are still visible on the display.



**To Display Required Files**



**To Display Specified Information**

**Figure 8-1. Example of How a Screen Display Can Be Customized**

#### **8.3.1.13 Consistency**

- Ensure that interactive control actions are consistent in form and consequence. Employ similar means to achieve similar ends, from one transaction to the next and from one task to another throughout the command and control system.
- Ensure that results of any control entry are compatible with user expectations, so a change in the state or value of a controlled element is displayed in an expected or natural form. For example, NEXT PAGE should call up the next page of the active file, not of an unrelated file.
- When selecting names for interactive control functions, choose names that are semantically congruent with natural usage, especially for paired opposites. For example, to move a cursor up, use UP. For the opposite command, use DOWN, not LOWER.

#### **8.3.1.14 Control**

- Allow the user to complete a control entry or action through an explicit action, such as ENTER, before the system interrupts to indicate a computer-recognized word.
- Ensure that control actions are simple, particularly for real-time tasks such as fire control that require rapid user response. Control logic should permit completion of a task with the minimum number of actions, consistent with user abilities.
- Allow the user to make control entries as needed, in essence, stacking commands.
- Ensure that the sequence of control entries is not slowed by delays in computer response. In general, system response time should be in the range of 5-50 milliseconds and no longer than 0.2 seconds. System response time, in this context, refers to the time between keystroke and screen response. It does not refer to response time for a query of a database.

#### **8.3.1.15 Feedback**

- Ensure that the system provides periodic feedback to indicate that normal operation is occurring if the user waits more than 15 seconds for the computer to respond.
- Ensure that the computer acknowledges every control entry immediately; for every action by the user, some reaction from the system should be apparent.
- When computer processing is lengthy in response to a control entry, provide an overt and positive indication of when processing has been completed.
- Provide displayed feedback for all user actions; display keyed entries stroke by stroke.

#### **8.3.1.16 Lockout**

- If application processing time requires a delay of concurrent user inputs and no keyboard buffer is available, lock out the keyboard until the computer is ready to accept the next input.
- When keyboard lockout has been terminated, provide a clear indication to the user.
- In situations where control lockout occurs, provide the user with a means of aborting the transaction that caused the lockout. A method such as a special function key can accomplish this transaction. The system should not reset and lose previous processing when aborted. The system should provide the option of resetting the system.

#### **8.3.1.17 Response Time**

- Ensure that the speed of computer response to user entries is appropriate to the type of dialog. Also ensure that responses are immediate to menu selections, function keys, and most entries during graphic interaction.
- Ensure that the speed of computer response to user control entries is appropriate to the transaction involved. Generally, those transactions perceived by a user to be simple should have faster responses.

#### **8.3.1.18 Pointer Design**

- Indicate the current pointer position by displaying some distinctive pointer symbol at that point. In all cases, try to obtain the highest contrast possible between the pointer and the background. Pointer size should be such that the pointer is not lost in the clutter of the background. A contrast ratio of 3:1 is the minimum recommended for an office environment.
- Provide the user with an easy, accurate means of pointing a displayed pointer at different display elements and/or display locations. The pointer positioning should work consistently throughout the application.
- For most graphics data entry, pointing should be a dual action, first positioning a pointer at a desired position, then confirming that position to the computer.

### **8.3.2 Context Definition**

#### **8.3.2.1 Application-Provided Context Definition to the User**

Design the interactive control of the application such that the user maintains an understanding of the context for the task being performed. Ensure that the system prompts expected user actions. For example, display the results of previous steps in the task affecting the present step and current options.

#### **8.3.2.2 Context Established by Prior Entries**

Design the interactive control software to interpret current control actions in the context of previous entries; do not require the user to re-enter data. Prompt the next logical action by the user.

#### **8.3.2.3 Record of Prior Entries**

Allow the user to request a summary of the results of prior entries (i.e., a history file) to help determine present status.

#### **8.3.2.4 Display Operational Mode**

When context for a user task is defined by the operational mode, display the current mode and any other pertinent information to the user.

#### **8.3.2.5 Consistent Display of Context Information**

Ensure information displayed to provide context for interactive control is distinctive in location and format and consistently displayed from one transaction to the next throughout all related applications.

#### **8.3.2.6 Highlighting Selected Data**

When a user is performing an operation on some selected display item, highlight that item.

#### **8.3.2.7 Display Control Parameters**

Allow the user to review any active control parameter(s).

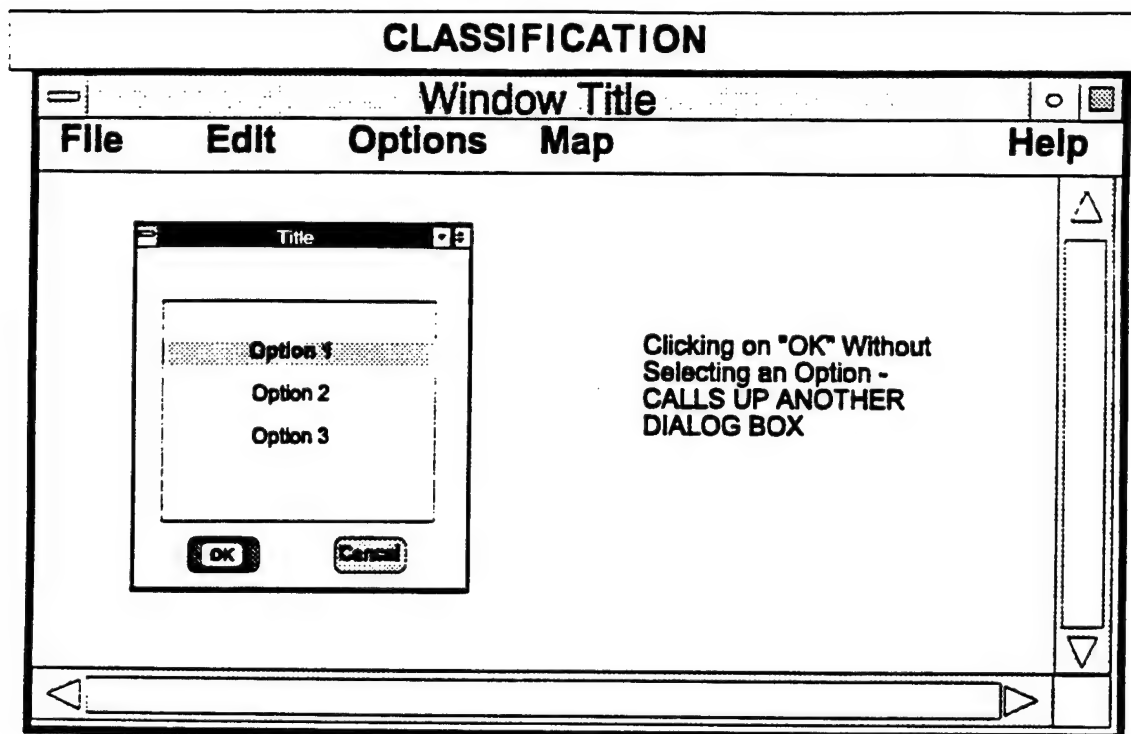
### **8.3.3 Transaction Selection**

#### **8.3.3.1 Consistent CONTINUE Option**

At any step in a defined transaction sequence, if there is only a single appropriate next step, provide a consistent control option, such as ENTER, to continue to the next transaction.

#### **8.3.3.2 Indicating Control Defaults**

When control is accomplished by keyed command or option code entries, and a default is defined as a null control entry, indicate that default to the user (see Figure 8-2).



**Figure 8-2. Example of How a System Can Display Default Status**

#### **8.3.3.3 User-Specified Transaction Timing**

When appropriate to task requirements, allow the user to specify transaction timing. For example, the user should be able to specify when a requested transaction should start, when the transaction should be completed, and/or the periodic scheduling of repeated transactions.

#### **8.3.3.4 Display Option Codes**

When the user must select options by code entry, display the code associated with each option in a consistent, distinctive manner.

#### **8.3.3.5 Available Options**

When it is desirable not to change the menu list, ensure that the user can clearly distinguish between available and unavailable options. Displaying unavailable options in a visually distinctive manner may aid navigation.



### **8.3.3.6 Stacked Control Entries**

Stacked commands are a function of command line interfaces. Allow the user to key a sequence of commands or option codes as a single stacked control entry. Stacked control entries, also called stacked commands, allow the user to input a series of command entries at one time. This may be done by continuous entry while the computer processes previous commands, or by typing in a series of commands and then entering them simultaneously.

- For control entry stacking, accept command names or their abbreviations or option codes, just as if those control entries had been made separately.
- Provide flexible transaction selection by allowing user to assign one name to a defined series of control entries. Use that named macro for subsequent command entry. Include a predefined informational query process (see Section 12.0) in all applications that provide a user database interface.
- For control entry stacking, require that entries be made in the order normally made when performing a succession of separate control entry actions.

### **8.3.3.7 Pointer Placement**

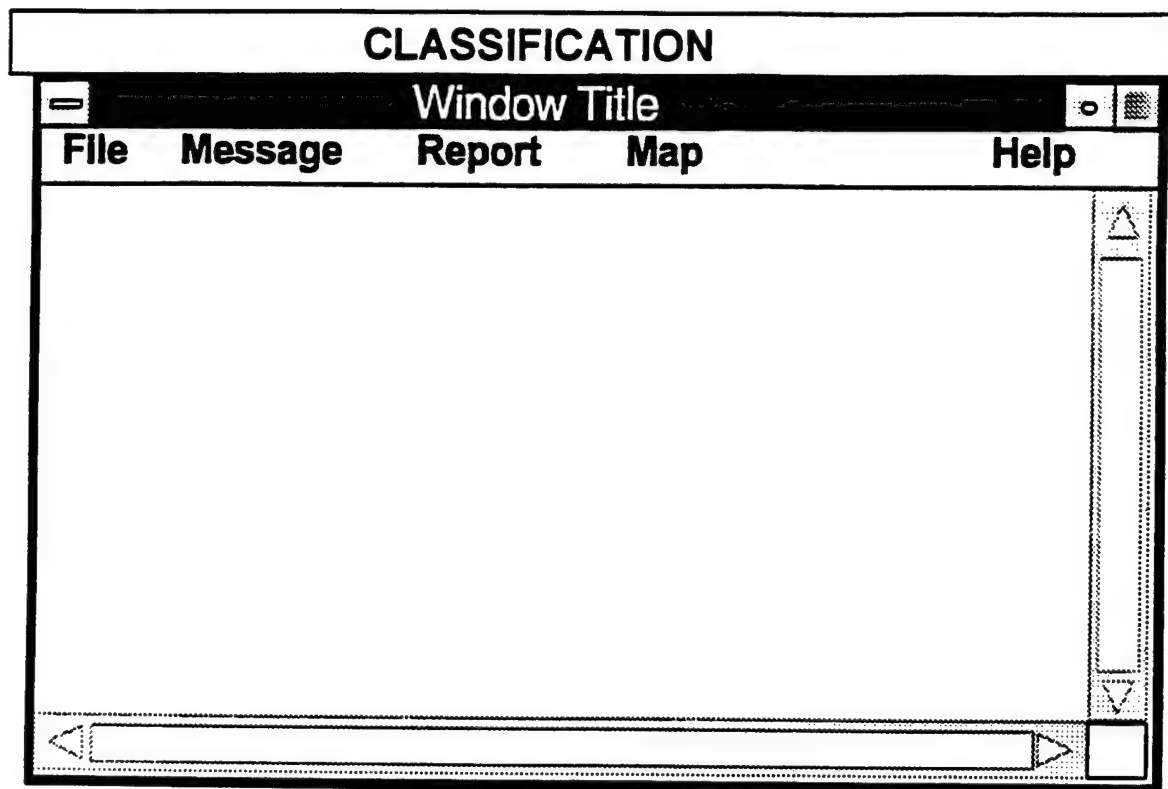
- When the user must select options by keyed entry of a corresponding code, place the pointer in the control entry area at display generation.
- When the user will need to select among displayed options by pointing, place the pointer on the first (most likely) option at display generation.

### **8.3.3.8 Prompting Control Entries**

Provide the user with whatever information may be needed to guide control entries. Incorporate prompts in a display at any point in a transaction sequence, and/or provide prompts in response to requests for HELP.

### **8.3.3.9 General List of Control Options**

Provide a general list of basic control options that will always be available to serve as a home base or consistent starting point for control entries. For an example, see Figure 8-3 (see also Paragraph 8.2.3.3).



**Figure 8-3. Example of a General List of Control Options**

#### **8.3.4 Interrupts**

The terms that follow in caps/bold represent functions that occur in many different styles. Use of these terms should be consistent with the style upon which the application is based. Those terms listed in this section are not intended as an exclusive list of terms.

##### **8.3.4.1 REVIEW Option**

If appropriate, provide a nondestructive **REVIEW** option that will return to the first display in a defined transaction sequence, permitting the user to review a sequence of entries and make necessary changes.

##### **8.3.4.2 PAUSE and CONTINUE Options**

If appropriate, provide **PAUSE** and **CONTINUE** options that will interrupt and later resume a transaction sequence without any change to data entries or control logic for the interrupted transaction.

#### **8.3.4.3 Indicating PAUSE Status**

If a PAUSE option is provided, display some indication of the PAUSE status whenever that option is selected by a user, and prompt the CONTINUE action that will permit resumption of the interrupted transaction.

#### **8.3.4.4 END Option**

If appropriate, provide an END option that will conclude a repetitive transaction sequence.

#### **8.3.4.5 Aborting or Escaping from a Function**

Ensure that the system makes it easy for the user to abort, escape, or exit from a current operation or function (see also Paragraph 8.3.4.9).

#### **8.3.4.6 Indicating System Status**

Inform the user that system action is continuing. Ensure that the “working” indicator has dynamic aspects to keep the user informed of continuing system function.

#### **8.3.4.7 SUSPEND Option**

- If appropriate, provide a SUSPEND option that will preserve current transaction status when a user leaves the system and permit resumption of work when the user later logs back onto the system.
- If a SUSPEND option is provided, display some indication of the SUSPEND status whenever a user selects that option. Prompt the user with those procedures that permit resumption of the suspended transaction at the subsequent log-on. For example, specifically prompt the user with “Type EXIT to return to application.”

#### **8.3.4.8 System Interruptions**

Ensure that the system interrupts the user only when necessary to prompt response, to provide essential feedback, and to signal errors.

#### **8.3.4.9 CANCEL Option**

If appropriate, provide a CANCEL option that will erase changes just made by the user and restore the current display to its previous version.

#### **8.3.4.10 Distinctive Interrupt Options**

If different types of user interrupts are provided, design each interrupt function as a separate control option with a distinct name.

#### **8.3.4.11 GOBACK Option**

If appropriate, provide a nondestructive GOBACK option that will display the previous transaction.

#### **8.3.4.12 RESTART Option**

If appropriate, provide a RESTART option that will cancel entries made in a defined transaction sequence and return to the beginning of the sequence. When data entries or changes will be nullified by restart, require the user to CONFIRM.

### **8.3.5 Error Management**

#### **8.3.5.1 User Confirmation of Destructive Entries**

When a control entry (including log-off) will cause extensive change in stored data, procedures, and/or system operation -- particularly if it cannot be easily reversed -- notify the user and require confirmation of the action before implementing.

#### **8.3.5.2 User Warned of Potential Data Loss**

Word the prompt for a CONFIRM action to warn the user explicitly of any possible data loss.

#### **8.3.5.3 Errors in Stacked Commands**

If an error is detected in a stacked series of command entries, ensure that the system either consistently executes to the point of error or consistently requires the user to correct errors before executing any command.

#### **8.3.5.4 Partial Execution of Stacked Commands**

If only a portion of a stacked command can be executed, notify the user and provide appropriate guidance to permit correction, completion, or cancellation of the stacked command.

#### **8.3.5.5 Flexible GOBACK for Error Correction**

Allow the user to GOBACK easily to previous steps in a transaction sequence in order to correct an error or make any other desired change.

#### **8.3.5.6 Explicit Entry of Corrections**

When the user has completed correcting an error, whether a command entry or data entry, require an explicit action to re-enter the corrected material. Use the same ENTER action for re-entry that was used for the original entry.

### 8.3.5.7 Prompting Command Correction

If an element of a command entry is not recognized or is logically inappropriate, ensure that the system prompts the user to correct that element, rather than require re-entry of the entire command.

### 8.3.5.8 Immediate Data Correction

If a data entry transaction has been completed and errors are detected, allow the user to make corrections directly and immediately.

### 8.3.5.9 Distinctive CONFIRM Action

Provide an explicitly labeled CONFIRM control, such as a function key or widget (e.g., control button, dialog box) different from the ENTER control, for user to confirm questionable or destructive control and data entries (see Figure 8-4).

### 8.3.5.10 UNDO to Reverse Control Actions

Ensure any user action can be immediately reversed by an UNDO command.

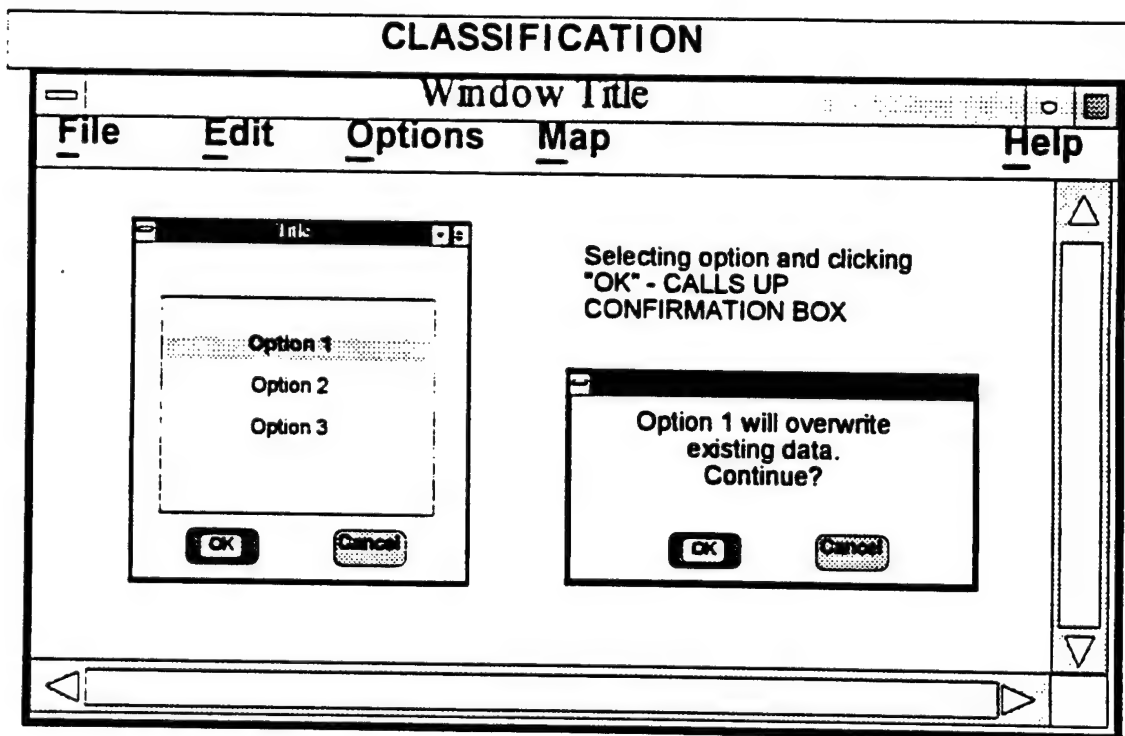


Figure 8-4. Example of a Distinctive Confirm Action, Using a Dialog Box

#### **8.3.5.11 Appropriate Response to All Entries**

Design software to provide an appropriate response for all possible control entries, correct and incorrect. For example, selecting an incorrect function key should cause a message indicating the appropriate selections.

#### **8.3.5.12 Appropriate Terms for All Entries**

Ensure software is consistent in the use of terms, and use only the most explicit term. Use "cancel" for cancel functions, rather than a simple acknowledgment such as "OK." Avoid complex terms (i.e., "Save & Apply" or "Exit to Prior Screen"), if possible. Ensure complex terms have one consistent meaning within an application.

#### **8.3.5.13 Display Duration**

Ensure notices, alerts, and informational displays remain visible to the user until responded to by specific user action. Field use of computers creates a situation where the user may not be continuously monitoring the screen presentation. Therefore, do not use automatic time-outs where mission-critical information is displayed.

#### **8.3.5.14 Selection Errors**

The pointing device interface uses both single and double clicks for control actions. Ensure that the software protects the system from inadvertent double clicks by the user and that the protection supplied is consistent with user and system requirements.

#### **8.3.5.15 Inappropriate Item Selection**

Cue the user to, but do not allow selection of, unavailable items. Do not allow output fields data entry without the user acknowledging selection of the option. Ensure that the software prevents data entry in any inappropriate field.

### **8.3.6 Alarms**

#### **8.3.6.1 Special Acknowledgment of Critical Alarms**

When the user must acknowledge special or critical alarms in a unique way, such as a special combination of key strokes, ensure this acknowledgment does not inhibit or slow the response to the condition initiating the alarm.

#### **8.3.6.2 Alarm Reset**

Provide the user with a simple means of turning off an auditory alarm without erasing any displayed message that accompanies the auditory signal. For noncritical alarms, provide a simple method for acknowledging and turning off the signal.

### **8.3.6.3 Distinctive and Consistent Alarms**

Ensure alarm signals and messages are distinctive for each class of event, such as INCOMING MESSAGE ALERT, TERMINAL STATUS, TRACK ALERT, etc.

### **8.3.6.4 Alarm Definition by User**

When monitoring tactical situations or tactical data status, allow the user to define the conditions (e.g., priorities, percentages, target flight path, etc.) that result in a software-generated alarm, alert, or status message.

## **8.4 FUNCTION KEYS**

The two types of function key are fixed and variable. The fixed key has only one predefined function associated with it. The variable key function will vary depending on the system mode or level within the interactive dialog. The function for the variable key is communicated to the user by changing the label located adjacent or internal to the key or through soft keys. Soft keys are objects on the display screen that represent the function keys on the keyboard. As the function of a key changes, the soft key labeling also changes. Fixed and variable function keys can be used together and with other dialog methods.

As with any interactive control method, the designer should note the following overarching design guidelines:

- Consistent design in terms of placement, labeling, and procedural logic
- Easy association with the function being called up through labeling located adjacent to the function keys
- Feedback
- Spatial consistency between the labeling and the function key.

### **8.4.1 General**

Function keys are located on the keyboard and activate a computer software function when pressed.

#### **8.4.1.1 Usage**

Consider function key dialog for:

- Frequently required control entries

- Tasks requiring only a limited number of control entries or in conjunction with other dialog types as a ready means of accomplishing critical entries that must be made quickly, without syntax error
- Interim control entries (i.e., for control actions taken before the completion of a transaction).

#### **8.4.1.2 Feedback for Function Key Activation**

When function key activation does not result in any immediately observable response from the computer, provide users with some other form of computer acknowledgment and feedback. No system function should be activated without an indication to the user.

#### **8.4.1.3 Disabling Unneeded Function Keys**

When function keys are not needed for any current transaction, temporarily disable those keys under computer control; do not require the user to apply mechanical overlays for this purpose. (see Section 8.4.1.7).

#### **8.4.1.4 Function Key Meaning**

In general, each function key should control only one function. If a key must control more than one function, display the actual or current meaning of the function to the user by displaying soft keys on the screen.

#### **8.4.1.5 Soft Key Design**

Locate soft function keys displayed on the screen close to the actual keyboard function keys and in the same spatial orientation. For example, on command and control system keyboards with function keys across the top, place soft keys at the bottom of the screen, directly above the keyboard as illustrated in Figure 8-5.

#### **8.4.1.6 Redundant Activation of Soft Key Function**

Enable the user to activate the function represented on a soft key through either the function key or a pointing device, such as a mouse.

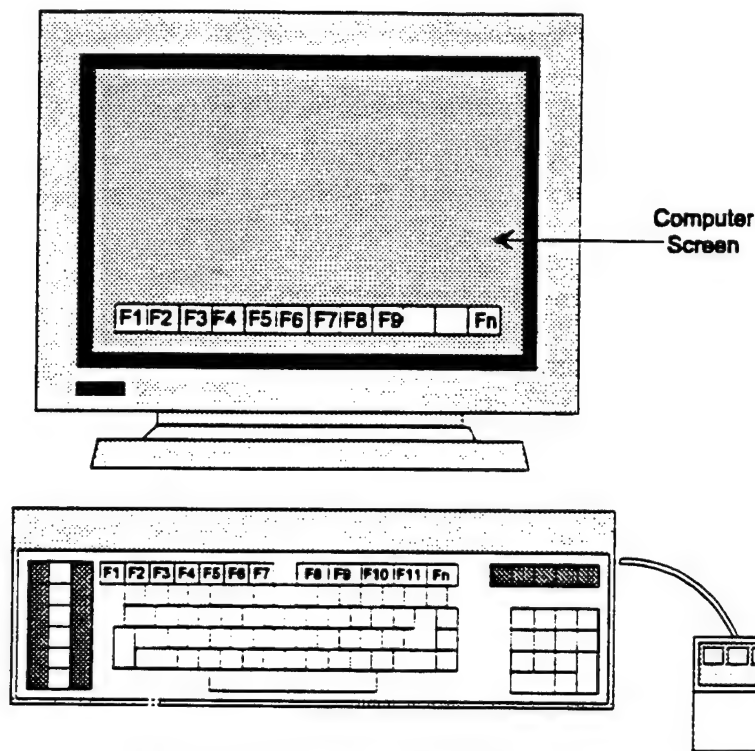
#### **8.4.1.7 Indicating Active Function Keys**

If some function keys are active and some are not, indicate the current subset of active keys in some noticeable way, such as brighter illumination or blanking of corresponding soft key labels on the display (see Figure 8-6).

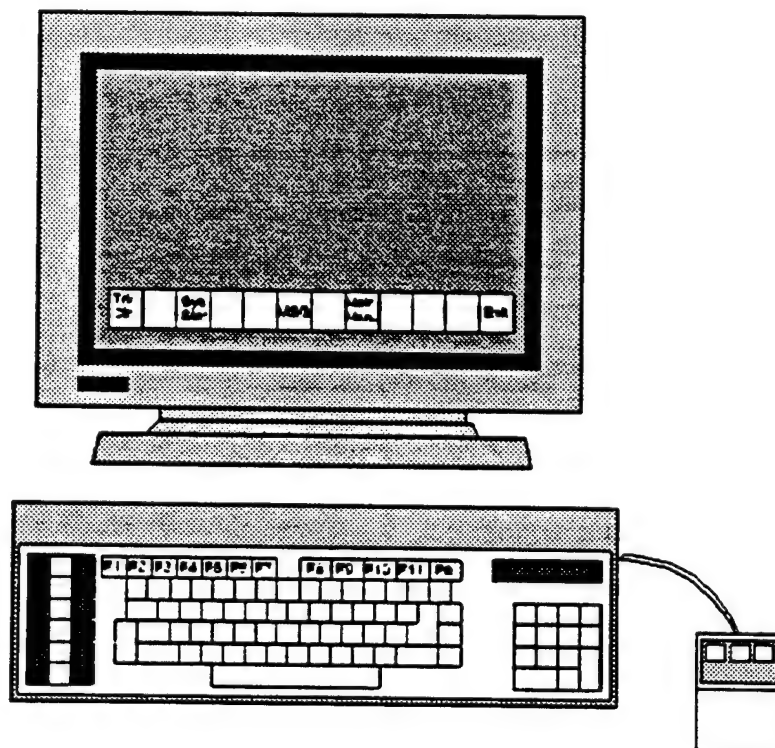
#### **8.4.1.8 Key Functionality Load**

Avoid overloading the functionality of keys. It is recommended that no more than two functions per key be used; however, provide the user with all necessary function controls required to perform the task.





**Figure 8-5. Example of Soft Key Location**



**Figure 8-6. Suggested Method for Indicating Active and Inactive System Function Keys**

#### **8.4.1.9 Easy Return to Base-Level Functions**

If user performs an action that changes the functions assigned to a key set, provide an easy means to return to initial, base-level functions or menu (see Figure 8-7).

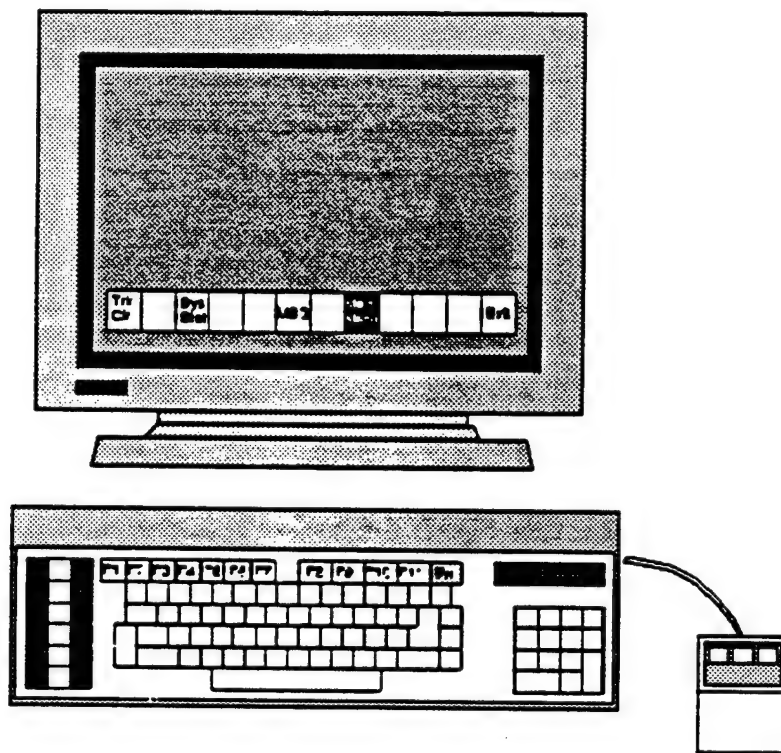
### **8.4.2 Consistency**

#### **8.4.2.1 Consistent Functions in Different Operational Modes**

When a function key performs different functions in different operational modes, assign equivalent or similar functions to the same key.

#### **8.4.2.2 Consistent Assignment of Function Keys**

If a function is assigned to a particular key in one computer transaction, assign that function to the same key in other transactions.



**Figure 8-7. Recommended Method for a Return to Base-Level Functions**

### **8.4.3 Double Keying**

#### **8.4.3.1 Logical Pairing of Double-Keyed Functions**

If double (control/shift) keying is used, the functions paired on one key should be logically related to each other.

#### **8.4.3.2 Consistent Logic for Double Keying**

If double (control/shift) keying is used, the logical relation between shifted and unshifted functions should be consistent from one key to another.

### **8.4.4 Labeling**

#### **8.4.4.1 Distinctive Labeling of Function Keys**

Label each function key informatively to designate the function it performs; make labels sufficiently different from one another to prevent user confusion.

#### **8.4.4.2 Labeling Multifunction Keys**

If a key is used for more than one function, always indicate to the user which function is currently available.

#### **8.4.4.3 Labeling of Menu Options for Function Keys**

When designing a command and control menu where options are selected through variable function keys, avoid using a function key number (e.g., F1, F2) as option designator. Instead, place the function key label just above the key on the display. See example in Figure 8-8.

### **8.4.5 Layout**

#### **8.4.5.1 Layout Compatible with Use**

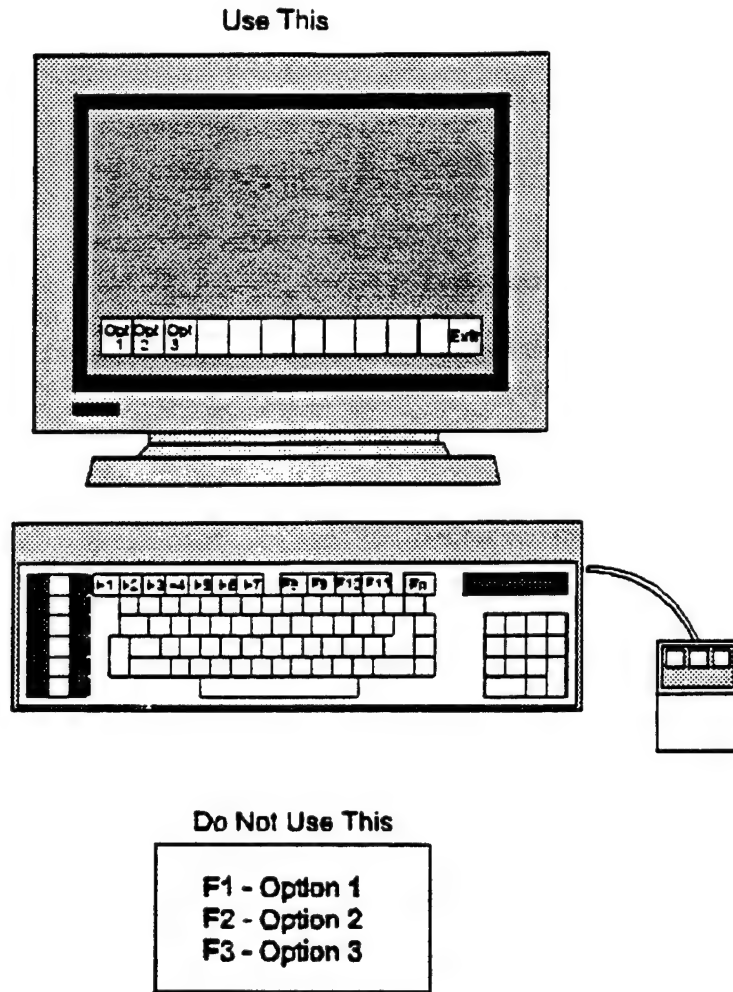
Make the layout of function keys compatible with their importance. Give keys for emergency functions a prominent position and distinctive coding (e.g., size and/or color).

#### **8.4.5.2 Safeguards**

Provide physical protection, software disabling, or interlocks for keys with potentially disruptive consequences.

#### **8.4.5.3 Distinctive Location**

Group function keys in distinctive locations on the keyboard to facilitate learning and use. Place frequently used function keys in the most convenient locations. For command and control systems, this should be at the top of the keyboard, just below the corresponding labels.



**Figure 8-8. Recommended Location for Function Key Labels**

## **8.4.6 Single Keying**

### **8.4.6.1 Single Activation of Function Keys**

Ensure that any key will perform its labeled function with a single activation and will not change function with repeated activation without indicating the new function or change in mode.

### **8.4.6.2 Single Key for Continuous Functions**

When a function is continuously available, assign that function to a single key.

### **8.4.6.3 Single Keying for Frequent Functions**

Keys controlling frequently used functions should allow single key action and should not require double (control/shift) keying.

## REFERENCES

Paragraph	References
8.0	Smith and Mosier (1986); Helander (1986); DoD (1989b)
8.1	DoD (1992a)
8.2.1	Kearsley (1988) p. 9
8.2.4.4	Horton (1990) p. 264
8.2.4.7	Brown (1988) p. 167, para 9.27
8.2.4.8	Hurd (1983) Cited in Horton 1990, p. 264
8.2.5	Walker (1987) Cited in Horton 1990
8.2.5.2	Relles and Price (1981) Cited in Horton 1990
8.2.5.3	Relles and Price (1981) Cited in Horton 1990
8.2.5.4	Relles and Price (1981) Cited in Horton 1990
8.2.5.5	Relles and Price (1981) Cited in Horton 1990
8.2.5.6	Relles and Price (1981) Cited in Horton 1990
8.2.6	OSF (1990) p. 8-2, para 8.1.1
8.2.6.1	OSF (1990) p. 8-2, para 8.1.1
8.2.6.6	OSF (1990) p. 8-3, para 8.1.3
8.2.7	Kearsley (1988) p. 79
8.2.7.1	Kearsley (1988) p. 79
8.2.7.2	Kearsley (1988) p. 79
8.2.7.3	Kearsley (1988) p. 79
8.2.9.1	DoD (1991) p. 7-3, para 7.3
8.2.9.2	Otte (1982) p. 273, para 3.7
8.2.9.3	Smith and Mosier (1986) p. 298, para 4.0.5
8.2.9.5	Smith and Mosier (1986) p. 302, para 4.0.17
8.2.9.6	Kearsley (1988) p. 68
8.2.9.6g	Fenchel (1981) Cited in Horton (1990)
8.2.10.1	Kearsley (1988) p. 76

## REFERENCES (cont'd)

Paragraph	References
8.2.10.2	Unix System Laboratories (1991) p. 4-27, para 3
8.2.12.1	Kearsley (1988) p. 25
8.2.13.1	Unix System Laboratories (1991) p. 4-27, para 3
8.2.13.7	Nickerson (1986); Smith and Mosier (1986) p. 302, para 4.0.17
8.3.1.1	Smith and Mosier (1986) para 3.0-9
8.3.1.2	Lickteig (1989) p. 9
8.3.1.3	Williams et al. (1987a) Appendix A p. A-2; Smith and Mosier (1986) para 8.3.7.2-2
8.3.1.4	Smith and Mosier (1986) para 3.0-12
8.3.1.5	Chao (1987) p. 360 and 361
8.3.1.6	Smith and Mosier (1986) para 3.0-17
8.3.1.7	Smith and Mosier (1986) para 3.0-7
8.3.1.8	Lickteig (1989) p. 35
8.3.1.9	Lickteig (1989) p. 34; Williams et al. (1987a) Appendix A p. A-3
8.3.1.10	Smith and Mosier (1986) para 3.0-8
8.3.1.11	Hamel and Clark (1986) p. 29; Smith and Mosier (1986) para 3.0-3
8.3.1.12	Williams et al. (1987b) Appendix A p. A-3
8.3.1.13a	Smith and Mosier (1986) para 3.0-6, 19
8.3.1.13b	Smith and Mosier (1986) para 3.0-16
8.3.1.13c	Smith and Mosier (1986) para 3.0-11
8.3.1.14a	Smith and Mosier (1986) para 3.0-5
8.3.1.14b	Smith and Mosier (1986) para 3.0-2
8.3.1.14c	Smith and Mosier (1986) para 3.0-19
8.3.1.14d	Smith and Mosier (1986) para 3.0-19; Salvendy (1987)
8.3.1.15a	Williams et al. (1987b) Appendix A p. A-3
8.3.1.15b	Hamel and Clark (1986) 3.0-20; Smith and Mosier (1986) para 3.0

## REFERENCES (cont'd)

Paragraph	References
8.3.1.15c	Smith and Mosier (1986) para 3.0-15; Baeker (1980); Hamel and Clark (1986) p. 30; Mallary (1985) p. 26; McCann (1983) p. 4; Slominski and Young (1988) p. 5
8.3.1.15d	Smith (1986) para 1.0-3
8.3.1.16a	DoD (1989a) para 5.15.4.1.1.1; Smith and Mosier (1986) para (1986) p. 30; DoD (1989b)
8.3.1.16b	DoD (1989a) para 5.15.4.1.1.2
8.3.1.16c	Smith and Mosier (1986) para 3.0-21; DoD (1989a) para 5.15.4.1.1.3
8.3.1.17a	Smith and Mosier (1986) para 3.1-2
8.3.1.17b	Smith and Mosier (1986) para 3.0-18;
8.3.1.18a	Bowser (1991) p. 6; Lewis and Fallesen (1989) p. 94; HFS (1988)
8.3.1.18b	Bowser (1991) p. 6; Lewis and Fallesen (1989) p. 94
8.3.1.18c	Lewis and Fallesen (1989) p. 94
8.3.2.1	Bowser (1991) p. 6; Smith and Mosier (1986) para 3.4-1; Bullinger et al. (1987) pp. 312-3; Hamel and Clark (1986) p. 26-27; Mallary (1985) p. 28, 9 p. 2, 14 p. 6, 3 p. 360
8.3.2.2	Smith and Mosier (1986) para 3.4-2
8.3.2.3	Smith and Mosier (1986) para 3.4-3
8.3.2.4	Smith and Mosier (1986) para 3.4-4
8.3.2.5	Smith and Mosier (1986) para 3.4-7
8.3.2.6	Smith and Mosier (1986) para 3.4-6
8.3.2.7	Smith and Mosier (1986) para 3.4-5
8.3.3.1	Smith and Mosier (1986) para 3.2-12
8.3.3.2	Smith and Mosier (1986) para 3.2-11
8.3.3.3	Smith and Mosier (1986) para 3.2-19
8.3.3.4	Smith and Mosier (1986) para 3.2-8
8.3.3.5	Smith and Mosier (1986) para 3.2-10

## REFERENCES (cont'd)

Paragraph	References
8.3.3.6	Smith and Mosier (1986) para 3.2-13
8.3.3.6a	Smith and Mosier (1986) para 3.2-15
8.3.3.6b	Bowser (1991) p. 6; Smith and Mosier (1986) para 3.2-18
8.3.3.6c	Smith and Mosier (1986) para 3.2-14
8.3.3.7a	Smith and Mosier (1986) para 3.2-7
8.3.3.7b	Smith and Mosier (1986) para 3.2-6
8.3.3.8	Smith and Mosier (1986) para 3.2-5
8.3.3.9	Smith and Mosier (1986) para 3.2-2
8.3.4.1	Smith and Mosier (1986) para 3.3-5
8.3.4.2	Smith and Mosier (1986) para 3.3-8
8.3.4.3	Bowser (1991) p. 7; Smith and Mosier (1986) para 3.3-7
8.3.4.4	Baeker (1980); Williams et al. (1987a) Appendix A p. A-2; Smith and Mosier (1986) para 3.3-1
8.3.4.6	Harrell (1987) p.5
8.3.4.7	Smith and Mosier (1986) para 3.3-3
8.3.4.8	Smith and Mosier (1986) para 3.3-2
8.3.4.9	Smith and Mosier (1986) para 3.3-4
8.3.4.10	Smith and Mosier (1986) para 3.3-6
8.3.5.1	Smith and Mosier (1986) para 3.5-7 and 3.5-11
8.3.5.2	Smith and Mosier (1986) para 3.5-8
8.3.5.3	Smith and Mosier (1986) para 3.5-4
8.3.5.4	Smith and Mosier (1986) para 3.5-5
8.3.5.5	Smith and Mosier (1986) para 3.5-13
8.3.5.6	Smith and Mosier (1986) para 3.5-6
8.3.5.7	Smith and Mosier (1986) para 3.5-3
8.3.5.8	Smith and Mosier (1986) para 3.5-12
8.3.5.9	Smith and Mosier (1986) para 3.5-9



## REFERENCES (cont'd)

Paragraph	References
8.3.5.10	Smith and Mosier (1986) para 3.5-10
8.3.5.11	Smith and Mosier (1986) para 3.5-1
8.3.5.12	Bowser (1991) p. 8
8.3.5.13	Bowser (1991) p. 7
8.3.6.1	Smith and Mosier (1986) para 3.6-5
8.3.6.2	Smith and Mosier (1986) para 3.6-4 and 3.6-3
8.3.6.3	Smith and Mosier (1986) para 3.6-2
8.3.6.4	Smith and Mosier (1986) para 3.6-1
8.4.1.1a	Smith and Mosier (1986) para 3.1.4-2
8.4.1.1b	Smith and Mosier (1986) para 3.1.4-1
8.4.1.1c	Smith and Mosier (1986) para 3.1.4-3
8.4.1.2	Bowser (1991) p. 9; Smith and Mosier (1986) para 3.1.4-10
8.4.1.3	Smith and Mosier (1986) para 3.1.4-12
8.4.1.4	Ziegler and Fähnrich (1988) p. 129
8.4.1.5	Ziegler and Fähnrich (1988) p. 129
8.4.1.6	Ziegler and Fähnrich (1988) p. 129
8.4.1.7	Smith and Mosier (1986) para 3.1.4-11
8.4.1.8	Nielsen (1987) p. 248; Bullinger et al. (1987) p. 309; McCann (1983) pp.3-4
8.4.1.9	Smith and Mosier (1986) para 3.1.4-16
8.4.2.1	Smith and Mosier (1986) para 3.1.4-15
8.4.2.2	Smith and Mosier (1986) para 3.1.4-14
8.4.3.1	Smith and Mosier (1986) para 3.1.4-7
8.4.3.2	Smith and Mosier (1986) para 3.1.4-8
8.4.4.1	Smith and Mosier (1986) para 3.1.4-4
8.4.4.2	Smith and Mosier (1986) para 3.1.4-5
8.4.4.3	Sidorsky (1994) p. 1.1-12

## **REFERENCES (cont'd)**

<b>Paragraph</b>	<b>References</b>
8.4.5.1	Smith and Mosier (1986) para 3.1.4-18
8.4.5.2	Smith and Mosier (1986) para 3.1.4-18
8.4.5.3	Smith and Mosier (1986) para 3.1.4-17
8.4.6.1	Smith and Mosier (1986) para 3.1.4-9
8.4.6.2	Smith and Mosier (1986) para 3.1.4-13
8.4.6.3	Smith and Mosier (1986) para 3.1.4

## 9.0 TEXT

Two topics will be addressed in Section 9.0. Subsection 9.1 addresses general topics unique to textual windows (i.e., data entry/update screens) that were not covered in Section 5.0, Windows. Subsection 9.2 addresses form filling as an interactive dialog. This approach to data entry requires little or no training and allows a relatively slow system response time. Applications primarily use form filling for completing standard message and data entry forms. As with any aspect of the HCI design, consistency of design is of paramount importance. The guidelines presented deal more with interactive control than with data entry. For more information on data entry, see Smith and Mosier (1986), MIL-STD-1472D (DoD 1989a), or DOD-HDBK-761A (DoD 1989b).

### 9.1 TEXTUAL WINDOWS

This section addresses general guidelines related to windows that are primarily textual (i.e., data entry/update screens).

#### 9.1.1 Data Field Labeling

In general, the appearance of the data should be pleasing to the eye with the arrangement uncluttered and functionality efficient. The following list of guidelines should help to achieve these objectives:

- Ensure that displays are not different from paper forms without justification; field ordering should be in logical sequence from the user's point of view.
- Ensure that the layout of data fields is consistent within an application, because one of the overall DoD architecture goals is to have consistency across all DoD applications in the layout of commonly used display (e.g., the "views" presented by applications for querying related databases).
- Ensure that data field labels are easily distinguishable from actual data. This distinction may be achieved using different fonts for labels and data or special characters as separators. For example, each label should be followed by a colon (:) and separated from the actual data by at least two spaces.
- Distinctly separate columnar data (at least three spaces between columns), with column headings displayed above the data and at least one row separating the column heading and the data.
- Ensure that labels are consistent throughout an application or set of applications.
- In ordinary use, ensure that field labels are protected and transparent to keyboard control so the cursor skips over them when spacing or tabbing.

- When a dimensional unit (e.g., \$) is always associated with a field, display it as part of the label so entry is not required by the user.

### **9.1.2 Updatable Fields**

Guidelines for data field updates follow:

- Distinguish updatable fields by underscores below the data field. If highlights or colors are also used, they should be the same throughout an application or set of applications.
- Cues should distinguish required from optional fields and should be consistent throughout an application or set of applications.
- When the length of a field is variable, the user should not have to right- or left-justify or remove blanks from the entered data.
- Ensure that the user is able to enter data in familiar units. The application should perform any required conversions (e.g., between geographic, geodetic, and Military Grid Reference System coordinates).
- Authorized personnel should be able to selectively inhibit updatable fields in a multi-field display. Such a feature would allow trainees to take on increasing database maintenance responsibilities as they learn. It also supports efficient on-line accomplishments of "mass changes" when batch updates are not available.

### **9.1.3 Text Cursor**

The purpose of the text cursor is to indicate to the user where entered data will be placed. The text cursor can be in any updatable input field. Guidelines for the text cursor follow:

- If the user clicks on a non-updatable field or anywhere on the form, the text cursor should not move.
- The text cursor should move between and within fields with the mouse or by using the Return/Enter key, Tab key, or the arrow keys.
- The cursor should not obscure the character displayed in the position it designates except for password and other non-display fields.
- When in insert mode, the text cursor should appear between the characters where the inserted text will be placed.
- When in overwrite mode, indicate (e.g., on status bar) that the current status of the application is in overwrite mode, or the text cursor should highlight the character that will be replaced.

## **9.2 FORM FILLING**

### **9.2.1 General**

Form filling is the method of interaction where the user enters a series of commands or data items in predefined, mandatory or optional fields.

#### **9.2.1.1 Usage**

- Use form-filling dialog as an aid for composing complex control entries.
- Use form-filling dialog as a means of displaying default values for the parameters in complex control entries.
- Use form-filling dialog for tasks where flexibility in data entry is needed (such as the inclusion of optional as well as required items), where users will have moderate training, and/or where computer response may be slow.

#### **9.2.1.2 Interrupts for Multiple Entries**

Where forms have multiple entries, provide the user GOBACK, CANCEL, and RESTART capabilities for editing the form prior to final input into the system (see Figure 9-1).

#### **9.2.1.3 Explicit Data Entry**

Data entry should be accomplished through an explicit action, such as pressing the ENTER key.

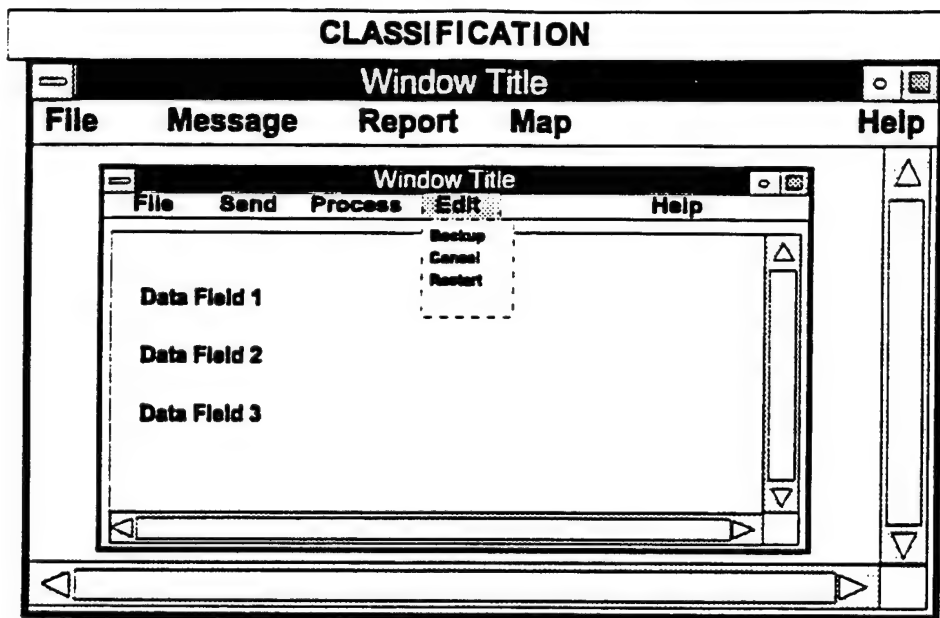
### **9.2.2 Defaults**

#### **9.2.2.1 Automatic Display of Default Data**

If default values are used in data entry fields, display them automatically in the appropriate data entry field.

#### **9.2.2.2 Replacement of Default Values**

When the user replaces a default value in a data entry field, ensure that the default definition is not changed.



**Figure 9-1. Example of Interrupt Capability for Multiple Entries**

### **9.2.3 Consistency**

#### **9.2.3.1 Consistent Format for Control Forms**

Ensure that forms for control entry are consistent in format.

#### **9.2.3.2 Format of Form and Hard Copy**

When the user enters data from hard copy into a computer, where possible, the computer form and hard-copy format should be identical (see Figure 9-2).

#### **9.2.3.3 Entry Dialog Consistency**

Dialog strategies for entering words and numbers should be consistent for a given set of logical functions throughout the system.

#### **9.2.3.4 Standard Formats**

Data and/or processes that have standard information requirements need to provide the standard format as part of the data screen. Message formats should include a template for the standard format. Using data entry screens that do not conform to user-accepted format will confuse users.

**REPORT**

**Data Field 1**

**Data Field 2**

**Data Field 3**

**Data Field 4                      Data Field 5**

**CLASSIFICATION**

Window Title

**File   Message   Report   Map   Help**

**REPORT**

**File   Send   Process   Edit   Help**

**Data Field 1**

**Data Field 2**

**Data Field 3**

**Data Field 4                      Data Field 5**

**Figure 9-2. Example of How a Paper Entry Form and a Computer Data Entry Form Should Be Consistent**

## **9.2.4 Cursor Movement**

### **9.2.4.1 Cursor Movement Into Non-Data Area**

Applications should not allow the user to move the cursor into a non-data-entry area during form filling.

### **9.2.4.2 Convenient Cursor Movement**

Ensure that the user has a convenient method for cursor control, such as the use of the tab or pointing device.

### **9.2.4.3 Cursor Movement by Explicit Action**

When moving from one data entry field to another, require the user to take an explicit action, such as hitting the tab control. The software should not automatically advance to the next field.

### **9.2.4.4 Cursor/Pointing Device Interaction**

Pointing device-to-cursor movement ratio should be close to 1:1. If appropriate, the user should be able to select the movement ratio.

### **9.2.4.5 Initial Cursor Location**

When the user first calls up a form, the cursor should be positioned in the first character space of the first data entry field (see Figure 9-3).

## **9.2.5 Data Field**

### **9.2.5.1 Variable Data Field Format**

For data entry fields with variable lengths, the software should automatically justify or truncate the data for the user. No leading characters should be required. See the example in Figure 9-4.

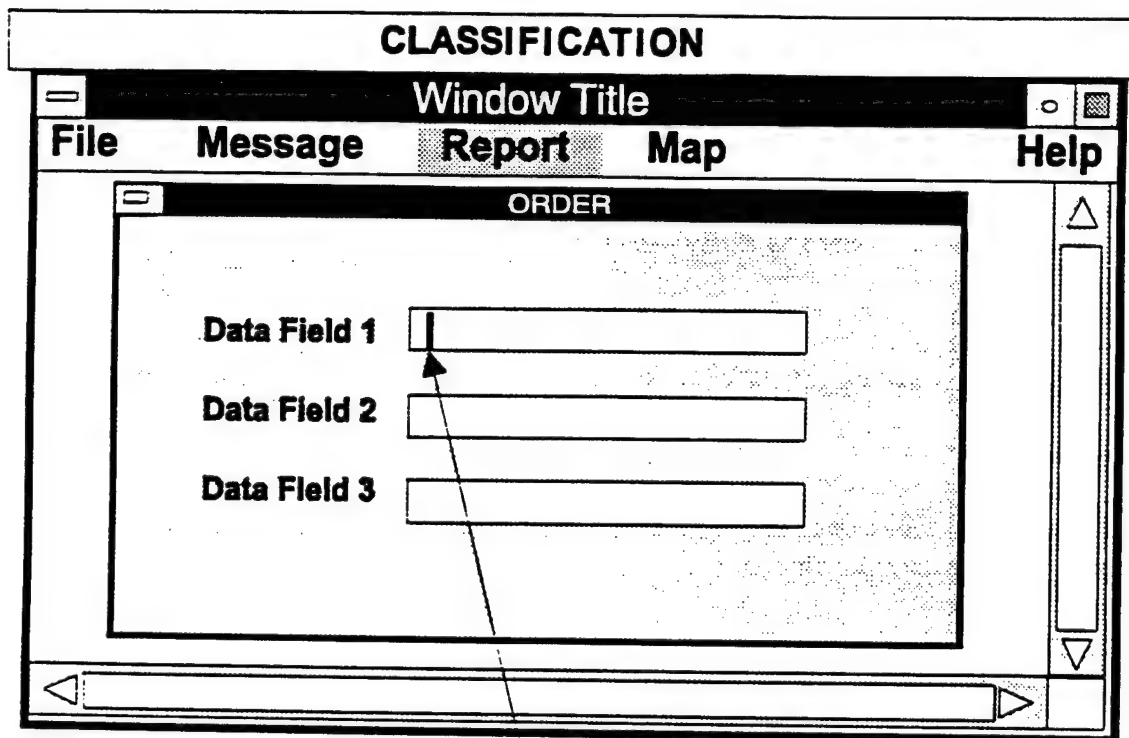
### **9.2.5.2 Consistent Format of Data Fields**

The format of data fields used frequently on different forms within and among applications should be consistent from one display to another and should use a format convention consistent with the user's expectations.

### **9.2.5.3 Subgroups Within a Data Field**

For data fields longer than 5 to 7 characters, break the field into subgroups of 3 to 4 characters that are separated by a space or delimiter. This should follow a convention consistent with the user's expectations (i.e., names, addresses, some descriptive information should not be subdivided).





**First Character Space  
Marked by Cursor**

**Figure 9-3. Cursor Should Appear in First Character Space of First Data Entry Field**

Enter This:	Not This:
73948	00073948

**Figure 9-4. Data Entry Should Not Require Leading Zeros**

#### **9.2.5.4 Data Field Boundaries**

Data fields should have distinctly marked boundaries.

#### **9.2.5.5 Data Field Identification**

Data entry fields should be clearly identified. Because the interface designs are often complex, users need a positive visual means to identify data entry fields.

#### **9.2.5.6 Field Length**

Data entry fields should be of fixed length, with cues given for their length (see Figure 9-5).

#### **9.2.5.7 Overwriting**

Data entry should not require overwriting of existing or default information. The field should either be empty, or the user should be required to perform an explicit control entry to erase the default data.

#### **9.2.5.8 Numeric Data Fields**

Numeric data in decimal format should use the decimal as part of the data display. Care should be taken to ensure the field size is adequate for the data range.

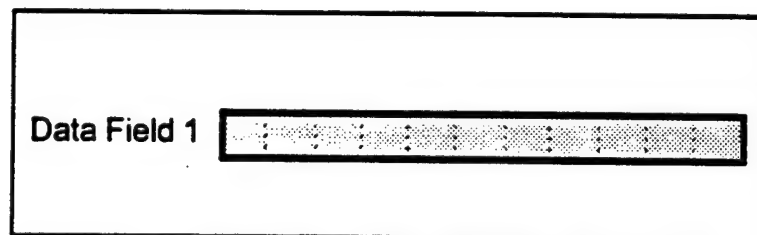
### **9.2.6 Error Management**

#### **9.2.6.1 Error Correction for Characters and Fields**

Ensure that the user can easily correct errors on a character-by-character and field-by-field basis.

#### **9.2.6.2 Error Messages**

Ensure that the software provides understandable error messages to the user when an unacceptable value is entered in a data field.



**Figure 9-5. Visual Cues for Field Length**

## **9.2.7 Form Layout**

### **9.2.7.1 Multiscreen Form Numbering**

If multiscreens are used for a transaction, provide page numbers for each screen. Also provide a means for rapidly returning to any page.

### **9.2.7.2 Logical Grouping of Data Fields**

Group related data fields together on the same form.

### **9.2.7.3 Explanatory Messages for Data Fields**

Provide explanatory messages for data fields that become visible when the cursor is placed in a field, when a user queries a field by clicking on the title, or by a context-sensitive help system. See the example in Figure 9-6.

### **9.2.7.4 Distinguishing Data Fields from Other Information**

Distinguish messages and instructions on a form from data entry fields by means of consistent location or other means of highlighting (see Figure 9-6).

### **9.2.7.5 Spacing and Boundaries**

Ensure that each data field has visible space and boundaries between it and other fields.

### **9.2.7.6 Grouping and Sequencing Fields**

Group and order data entry fields should be grouped and ordered on the form in a way that is logical for the task to be performed. This can be by sequence, frequency, or importance.

### **9.2.7.7 Form Title**

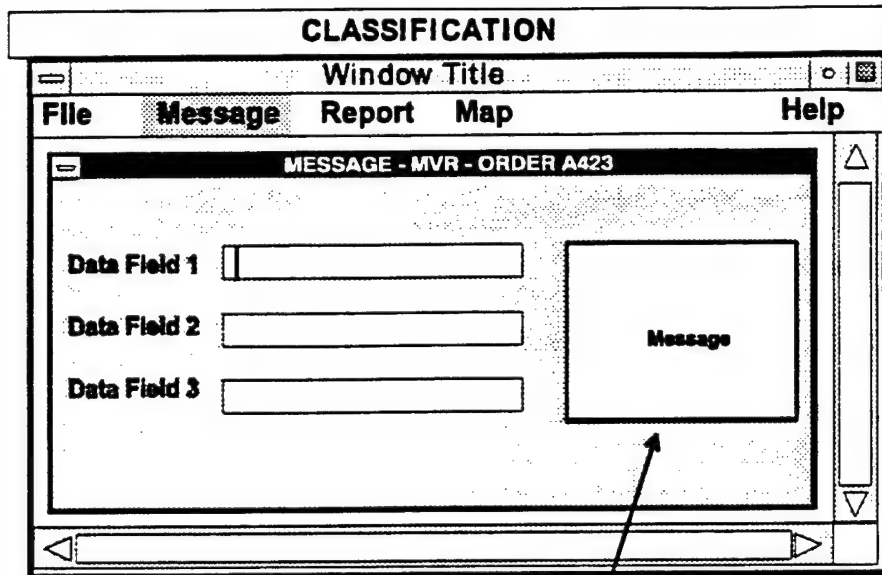
Ensure that each form-filling dialog display page has a meaningful title located at the top of the form, as illustrated in Figure 9-7.

### **9.2.7.8 Optional Field Labels**

Optional fields should be labeled or coded in a readily apparent manner (see Figure 9-8).

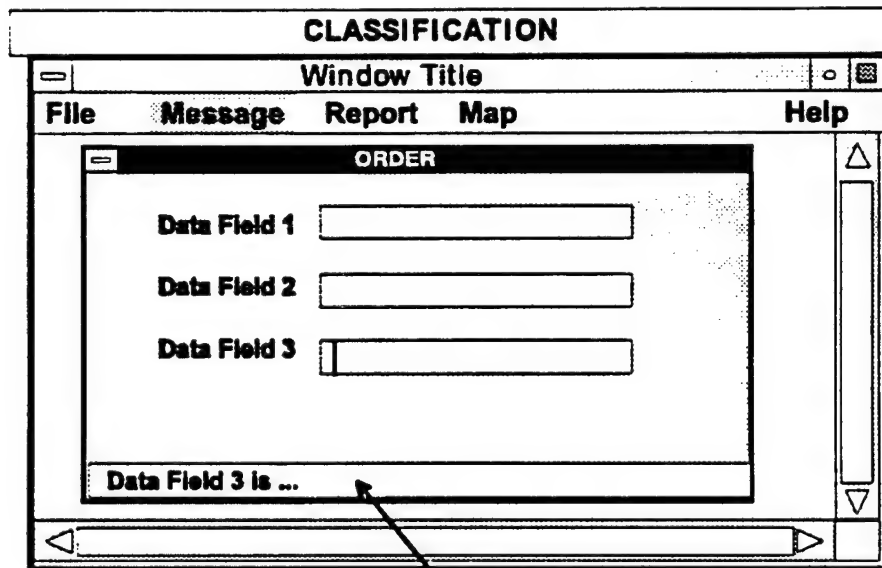
### **9.2.7.9 Optional Field Defaults**

When a data entry field in a form is optional, any value displayed in that field should be a default value.



User places cursor on title, clicks,  
information on field appears as a pop-up  
subwindow

OR



Information on data field appears automatically  
when cursor is placed in field.

**Figure 9-6. Example of How Explanatory Messages Can Be Provided**

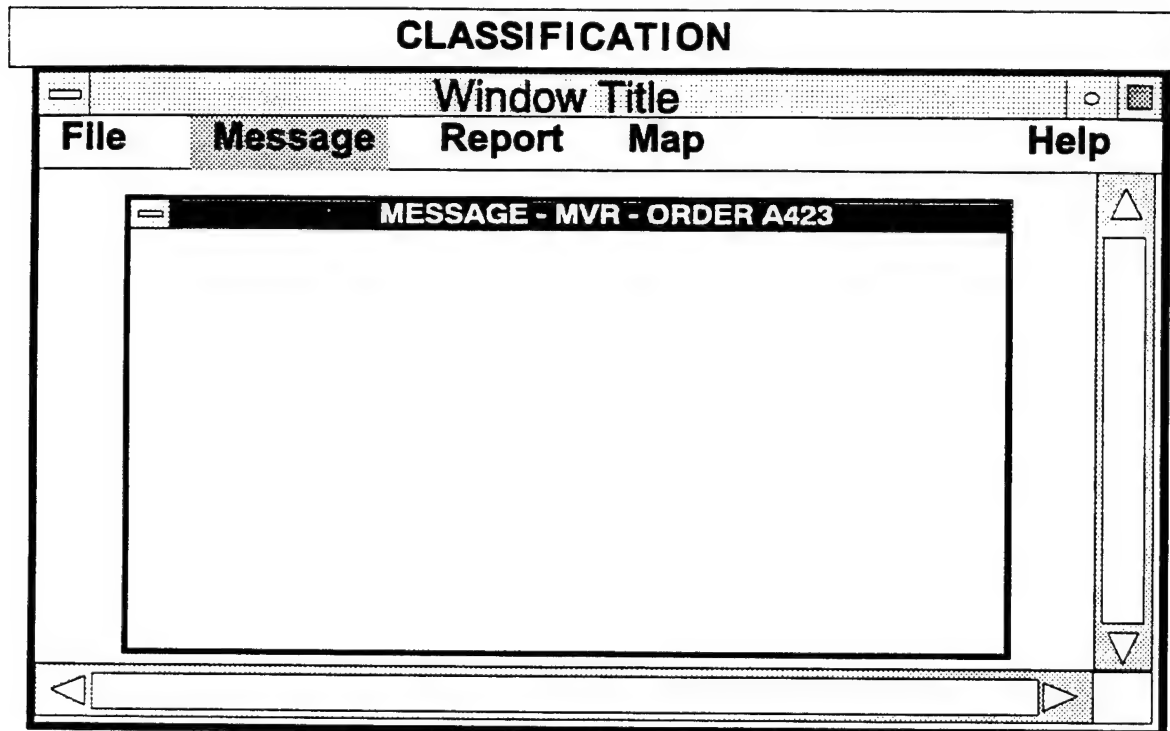


Figure 9-7. Example of a Form Title

Figure 9-8. Example of an Indication of an Optional Field

#### **9.2.7.10 Mandatory Fields**

The software application should not allow the user to bypass a mandatory field without data entry (see Paragraph 9.2.5.3).

### **9.2.8 Labeling**

#### **9.2.8.1 Distinctive Labeling**

Data fields, unless similar or identical, should have distinctive, explicitly descriptive labels.

#### **9.2.8.2 Data Field Label Location**

Application data entry field labels should be located either directly to the left or above the actual entry field and separated by at least one character.

#### **9.2.8.3 Similar Data Field Labeling**

Similar data entry fields should be labeled and located consistently for all forms.

#### **9.2.8.4 Consistent Labels**

Labels and instructions should be consistent from one application to another within related applications and to the extent possible across all systems.

#### **9.2.8.5 Field Label Familiarity**

Labels for data fields should be composed of terms familiar to the user and the task to be performed.

#### **9.2.8.6 Understandable Labeling**

Labeling for data fields and instructions should be easily understood by the typical user.

#### **9.2.8.7 Units of Measure**

Units of measure should be part of the data entry field label. If measurement units can change, this portion of the label should change automatically when new units are selected.

#### **9.2.8.8 Blanks Versus Nulls**

There should be a visible distinction between blanks and nulls in a data field.

## REFERENCES

Paragraph	References
9.1	DoD (1992a)
9.2.1.1a	Smith and Mosier (1986) para 3.1.2-2
9.2.1.1b	Smith and Mosier (1986) para 3.1.2-3
9.2.1.1c	Smith and Mosier (1986) para 3.1.2-1
9.2.1.2	Chao (1986) p. 13
9.2.1.3	Chao (1986) p. 13
9.2.2	Chao (1986) p. 13
9.2.3.1	Smith and Mosier (1986) para 3.1.2-4
9.2.3.2	Sidorsky (1984) p. 1.1-11; Chao (1986) p. 13
9.2.3.3	Lewis and Fallesen (1989) p. 8; Chao (1986) p. 13
9.2.3.4	Bowser (1991) p. 8
9.2.4.1	Chao (1986) p. 13
9.2.4.2	Shneiderman (1988) p. 702
9.2.4.3	Chao (1986) p. 13
9.2.4.4	Chao (1986) p. 13
9.2.5	Chao (1986) p. 13
9.2.5.9	Bowser (1991) p. 8
9.2.6	Shneiderman (1988) p. 702
9.2.7.1	Chao (1986) p. 13
9.2.7.2	Chao (1986) p. 13
9.2.7.3	Chao (1986) p. 703
9.2.7.4	Chao (1986) p. 13
9.2.7.5	Shneiderman (1988) p. 702
9.2.7.6	Shneiderman (1988) p. 702
9.2.7.7	Shneiderman (1988) p. 702
9.2.7.8	Shneiderman (1988) p. 703; Chao (1986) p. 13

## REFERENCES (cont'd)

Paragraph	References
9.2.7.9	Chao (1986) p. 13
9.2.7.10	Avery et al. (1990) p. 3-19
9.2.8.1	Chao (1986) p. 13
9.2.8.2	Chao (1986) p. 13
9.2.8.3	Chao (1986) p. 13
9.2.8.4	Shneiderman (1988) p. 702
9.2.8.5	Shneiderman (1988) p. 702
9.2.8.6	Shneiderman (1988) p. 702
9.2.8.7	Chao (1986) p. 13
9.2.8.8	Chao (1986) p. 13



## 10.0 GRAPHICS

Graphical presentation of data is a critical feature of many emerging DoD applications. This section provides guidelines for presenting data in graphical formats. The applications discussed here include tactical graphics (overlays, symbology, and terrain representation), pictographic representations (digitized maps, pictures, etc.), and presentation graphics (charts and graphs). Guidelines pertaining to graphical characteristics of the user interface (e.g., screen design, windows, icons, buttons, etc.) are presented in other sections of this document.

Most of the guidelines presented in this section were obtained from Lewis and Fallesen (1989) and Smith and Mosier (1986), who included information gathered from relevant Military Standards and other key documents. Additional guideline materials were obtained through literature reviews.

Subsection 10.1 focuses on map graphics. The designer of map graphic displays should note the following overarching guidelines that are relevant to electronic map displays:

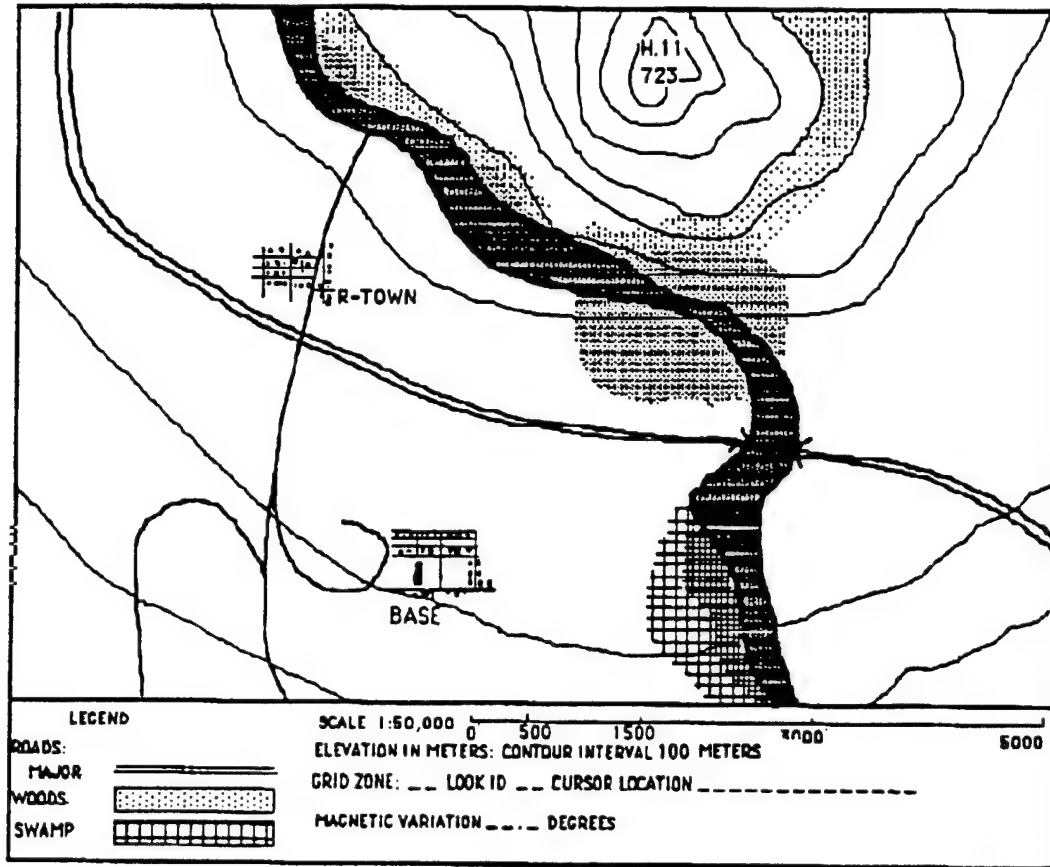
- The design of maps, including the use of symbology, should be consistent with the user's expectations.
- The level of detail should be consistent with the operational need. Too much or too little detail limits the usefulness of the map.
- Map graphics should have tools built in that allow the user to move easily around the map, to include zooming, panning, insets, registration, and keys for scale.

Subsection 10.2 focuses on presentation graphics. The goal of presentation graphics is to communicate effectively to the user. The idea, information, or concept communicated should be clear and unambiguous when presented in a visual form; otherwise alternate communication modes should be used. Some emerging technological capabilities allow the direct manipulation of elements of graphic objects within an application. These capabilities should be included in applications designed to interactively create graphics. The *Style Guide* will address three aspects of presentation graphics: graphs, pictures, and diagrams.

### 10.1 MAPS AND SITUATION DISPLAYS

#### 10.1.1 General

Maps refer to projected representations of geographic data, usually on flat surface displays. Maps include both natural and man-made features and text and/or graphics and colors used to describe or code those features. Situation displays provide a means of relating changing conditions or events to geographic features represented on maps. Figure 10-1 illustrates a typical map graphic display.



**Figure 10-1. Typical Electronic Map in Black and White**

#### **10.1.1.1 Curvature**

Be consistent in projecting the earth's curvature on flat surface maps when displaying large geographic areas. Provide the user with a method of determining the type of map projection used.

#### **10.1.1.2 Situation Display Presentation**

Provide a means of presenting situation displays as overlays on related map backgrounds.

#### **10.1.1.3 Map Label Position**

Position map labels consistently (e.g., beneath or within the feature). Label all significant features without cluttering the display, where possible.

#### 10.1.1.4 Map Orientation

Use a consistent map orientation when more than one map will be displayed (e.g., north consistent for all maps). It is recommended that all maps be north-oriented and the north direction annotated (see Figure 10-2).

#### 10.1.1.5 Designating Map Areas

Consider using color, shading, texture patterns, or highlighting to define map areas of special interest. Shades (tones) of a single color are preferable to multiple colors when observers must make relative comparisons between or among areas. When using shades of color or texture patterns, the gradation of shades from dark to light should correspond to variation in the variable that is represented (see Paragraph 4.3.3).

#### 10.1.1.6 Automated Tools

Provide automated tools for complex map analyses. The specific tools should be based on the user's needs. For example, avenue of approach, line-of-sight, and trafficability are needed by some but not all users. Determine user requirements, and provide appropriate tools.

#### 10.1.1.7 Selectability

Enable the user to select a single item within a densely packed group. When a graphics item is selected, highlight it.

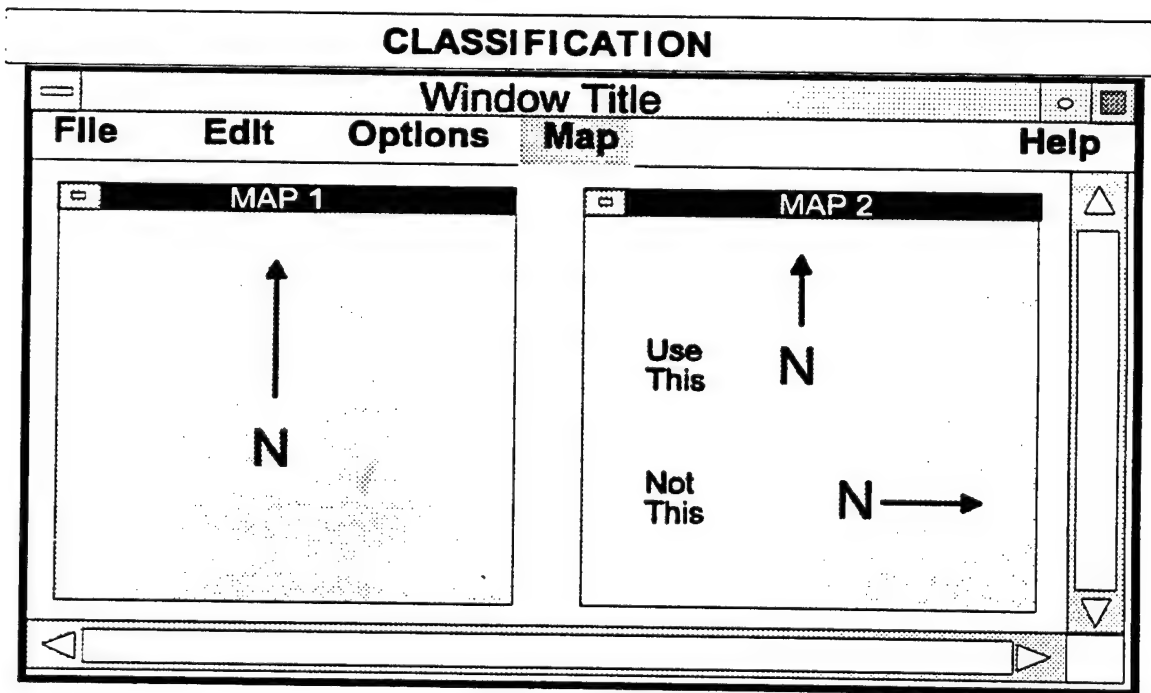


Figure 10-2. Example of Consistent Map Orientation

## 10.1.2 Static Display Attributes

### 10.1.2.1 Coverage Area and Resolution

As a minimum, ensure that maps cover user areas of responsibility at each organizational level, and provide all essential details required to conduct operations. Map displays should be large enough to permit the simultaneous presentation and visual integration of information required by the user. Small electronic displays may be panned and zoomed to increase map coverage. See Paragraph 10.1.3. However, at present, such displays have significant visual limitations when compared to traditional, large-format, paper maps.

- Ensure that all critical map features are represented.
- Ensure that labels remain legible at all display resolutions.
- Provide a means for reducing clutter while preserving essential information.
- Given a land-based command and control application, enable maneuver commanders at each echelon to view their own areas of operation, activities one echelon above and two echelons below, and activities of friendly adjacent (flanking) units. Also, display the activities of adjacent and deep enemy units that oppose displayed friendly forces (see Figure 10-3).

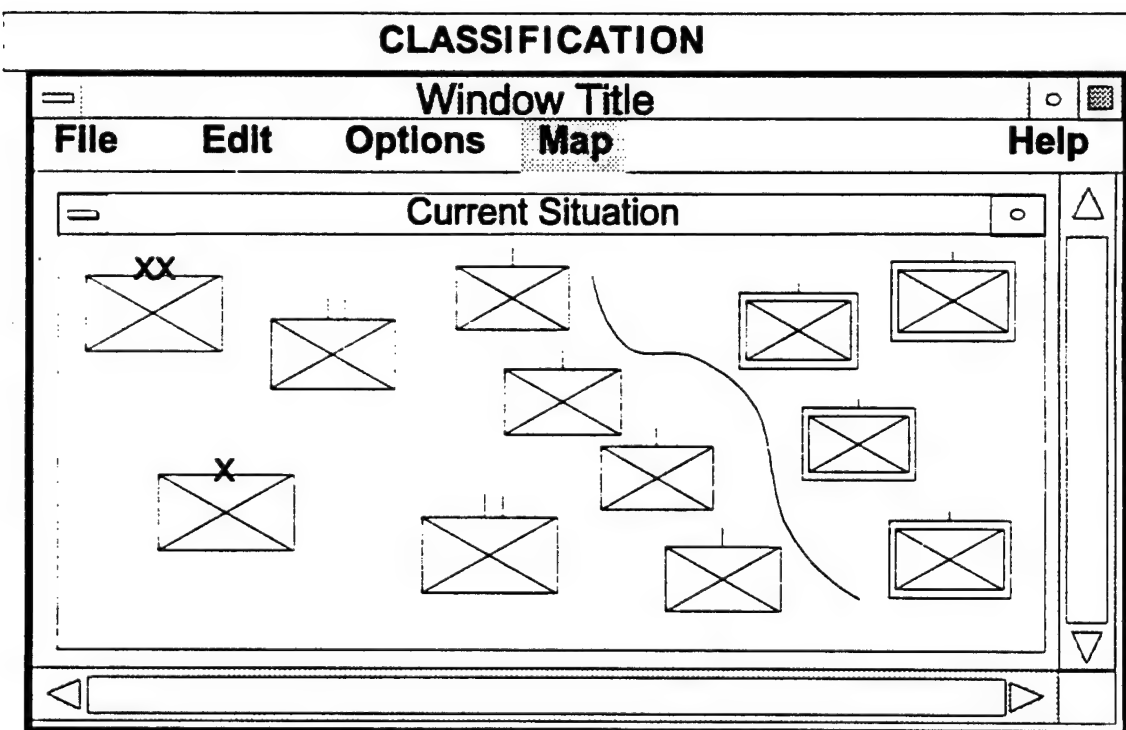


Figure 10-3. Brigade's Map Overlay Showing One Echelon Higher and Two Lower

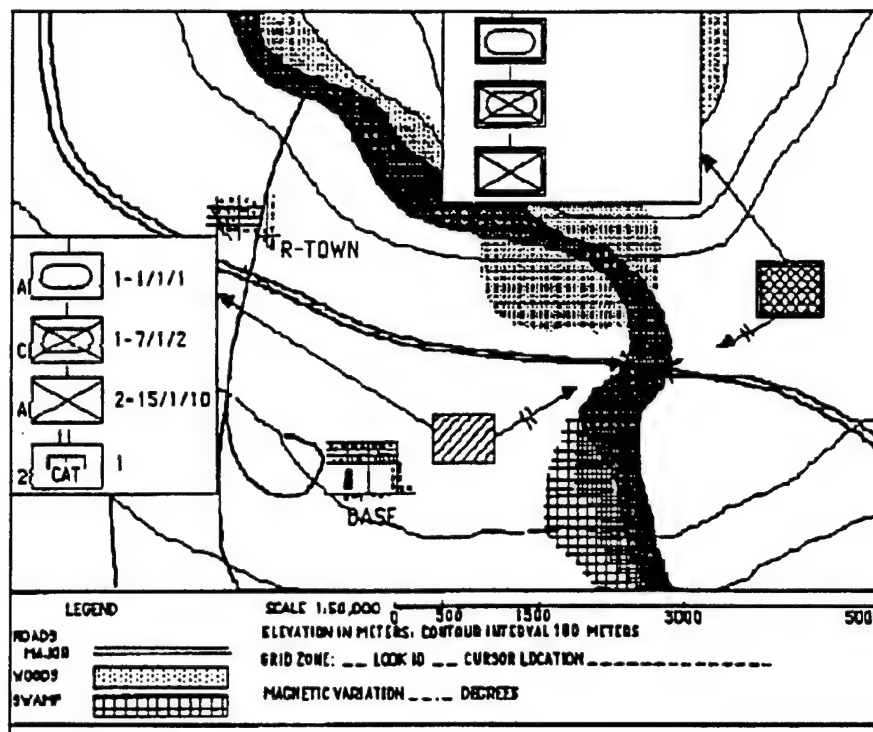
### 10.1.2.2 Accuracy of Location

- Place symbols accurately on the map or connect to the desired location using arrows, lines, or other pointing graphics.
- Provide an automated means of registering graphic data with background map information at all display scales.

### 10.1.2.3 Symbology

Ensure that colors, symbols, line size/quality, and fonts are consistent throughout a given system. When possible, display symbology should conform to published standards (e.g., Army Field Manual 101-5-1 [1985b], NATO Standardization Agreement 2019 [1990], or the DIA Standard Military Graphics Symbols manual, 1990 draft), but each system should also be able to use a commercial graphics editor to accommodate the creation and display of system-unique features and symbols. The following guidelines are recommended:

- Use standard military symbols in accordance with doctrine when preparing maps and overlays. For example, the Army should use the current edition of FM 101-5-1, *Operational Terms and Symbols*.
- Provide a means by which the user may obtain help in identifying unknown symbols or other map information. For example, the user could highlight a symbol and query its meaning through a context-sensitive help feature.
- Use standard military map color codes, and provide a user-prompted key defining the color codes that are used (see Subsection 4.3).
- Do not allow map symbols to overlap, particularly if this would obscure their identity. Provide a means for moving background symbols to the foreground or otherwise revealing masked symbols where overlap is unavoidable.
- Display essential labels (e.g., unit identification) with the symbol; otherwise, provide a means by which the user can display information related to selected symbols. Figure 10-4 illustrates how a user could query a symbol for more detail.
- Consider the auxiliary use of alphanumeric coding where graphic data are not already so labeled.
- Position symbol labels consistently in accordance with doctrinal guidance.



**Figure 10-4. Querying a Summary Symbol for Detailed Information**

- Digital terrain and elevation data (DTED), available from Defense Mapping Agency (DMA) for some versions of electronic map (e-map), provide information that allows alternative methods of portraying terrain features. In addition to traditional topographic contour intervals, DTED can provide data for map overlays depicting road networks, drainage, vegetation, and soil type. Use shading, coloring, or other visual cues to accentuate terrain features.

#### **10.1.2.4 Location of Displayed Section**

Display a constantly visible display of coordinates associated with the cursor in user-selectable coordinate units, which can be changed conveniently where location information is often used. Augment continuous display of location with the capability to fix (point on map) a location to facilitate moving overlay displays. The coordinate display should be capable of displaying multiple coordinate units concurrently.

- Provide to the user a means of obtaining the exact map coordinates for a selected symbol or map feature by means of querying the symbol or feature. The recommended method of querying an item is to use a pointing device to place the cursor on the graphic to be queried and “click” the pointing device.
- When the entire map is not displayed, provide an inset that shows where the displayed portion is located within the larger map (see Figure 10-5).

- Provide an automated means for readily determining the distance between points.
- Provide a means for readily determining the bearing between points.

#### 10.1.2.5 Area Bounding Boxes

Use bounding boxes when displaying maps in the main graphics drawing area. Area bounding boxes are pairs of coordinates defining a rectangular area, for example, latitude and longitude. Display the bounding coordinates for the geographic area being shown.

#### 10.1.3 Dynamic Characteristics

In a map graphics application, make functions available through menus to permit the user to make measurements, perform analysis, and control the appearance of the display. A method is needed to scan and change the scales of the maps because of the limited screen size of many displays. In addition, changes in the tactical situation require updates to various map overlays. The following guidelines should be considered when implementing dynamically changing maps.

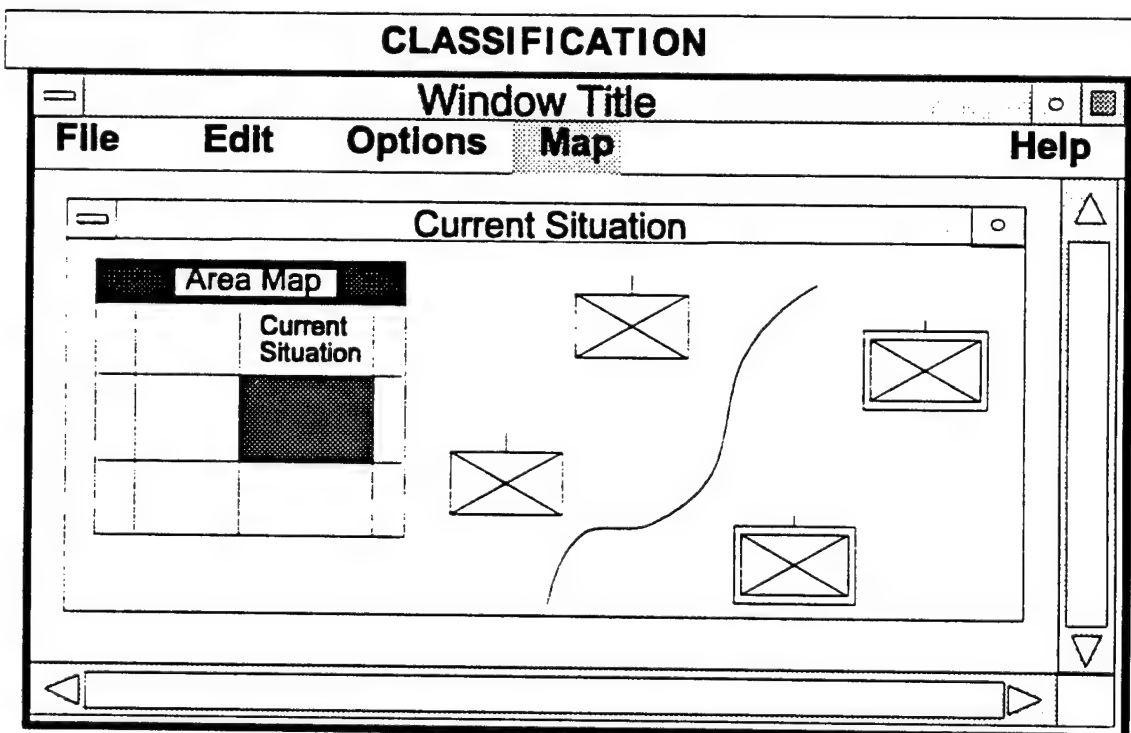


Figure 10-5. Example of a Map Inset

#### **10.1.3.1 Panning**

- Permit the user to change the displayed area by moving a window over the map in any direction. Panning operations may be continuous (preferable) or discrete but should meet the user's requirements.
- During panning operations, provide an indicator of position in the overall display.
- During panning operations, provide a means for rapidly returning to the starting point.

#### **10.1.3.2 Zooming**

- Provide a means for moving away from or toward the displayed area (zooming) to obtain a larger view or greater detail.
- Ensure that zooming does not cause problems in reading symbols, labels, or other map features.
- It is recommended that the level of detail (number of symbols and features depicted) be modified to match the degree of zooming used (i.e., more detail for close-up views and less for large-area perspectives).
- Of the two methods of zooming (i.e., continuous and discrete), continuous is preferable. Ensure that the method used is satisfactory to the user.
- When zooming, collapse symbols into fewer summary symbols to declutter.
- Provide a means for quickly returning to the normal display size when zooming.
- When changing scales through zooming, provide an indicator that continually shows the appropriate scale.
- It is recommended that an inset or window be provided that shows the maximum available map coverage. An example of map coverage (see Figure 10-5) would be a graphic square on the inset map that indicates the position of the map currently displayed. In the most useful form, this inset would be interactive and used to set parameters for calling up a screen map display.

#### **10.1.3.3 Automatic Updating**

Automatic updating, editing, and distributing map data are among the primary advantages offered by electronic displays. The following guidelines address considerations in implementing these capabilities:

- As appropriate, allow the user to select categories of information that will be automatically updated.



- Provide stable reference elements (e.g., terrain features, boundaries, etc.) when displays are automatically updated.
- Provide a means for readily identifying updates or changes. Critical changes must be easily recognized and distinguishable from other changes to the display. For example, highlight the update until the user acknowledges it.
- Allow the user to control how often the display is updated and to freeze the display to prevent further updates.
- Ensure that the rate of display update matches the perceptual abilities of the observer to permit successful visual integration of the changing patterns.
- Permit the user to freeze the display to prevent further updates. Provide a warning while the automatic display updating is suspended and when resuming automatic updating. Provide an option to either resume at the current time or at the time updating was suspended.

#### **10.1.3.4 Sequencing**

Display sequencing may be used to reduce clutter (e.g., presenting map overlays in succession), to reproduce temporal changes in the display database (e.g., changes in the tactical situation), and to aid in visualizing simulated changes in the battlefield situation.

- Allow the user to control the rate of sequencing where possible.
- Provide a capability to pause or suspend sequencing operations and provide an indicator of the status of sequencing operations.
- Allow the user to present sequenced displays in forward or reverse order as appropriate.
- Provide a means for the user to return quickly to a selected display within a sequence of displays.
- Consider using animation as an aid to the pictorial display for complex objects.

#### **10.1.3.5 Grid Overlay**

Provide a user-selectable grid overlay that is keyed to the coordinate system of the map. It should be easy for the user to turn the grid on and off. Coordinate keying of the overlays must be clearly specified and easily operated by the user.

#### **10.1.3.6 Dynamic Map Legend**

Ensure that the map display has an associated window giving relevant information in a continuous display. The information should include map scale, cursor location, graphic of map coverage, and status (i.e., "working," "computing," "available," etc.).

#### **10.1.3.7 Cursor Design**

Ensure that the cursor includes a point designation feature (e.g., cross hairs or a v-shaped symbol), because fine accuracy is often required in positioning the cursor.

#### **10.1.3.8 Distance/Azimuth**

Provide a distance/azimuth function that calculates the distance (range) and azimuth (bearing) between any two selectable points or symbols. Present distance in selectable units (feet, meters, miles, or kilometers). Azimuth should be displayed in degrees from true north.

#### **10.1.3.9 Position Determination**

The “determine position” function calculates the position of the point that is identified, and the answer should be presented in a selectable coordinate system (e.g., Universal Transverse Mercator, latitude/longitude, or Military Grid Reference System). It is recommended that answers be provided textually in user-specified units of measure, such as latitude and longitude, distance (in nautical miles), and azimuth (in degrees from true north).

### **10.1.4 Creating And Editing Map Graphics**

#### **10.1.4.1 Standard Symbol Library**

Provide a library of standard symbols and a means of copying and manipulating symbols.

#### **10.1.4.2 Labeling Symbols**

Provide an easy means of labeling symbols. Consider automated means of aiding the user in labeling and enforcing labeling conventions.

#### **10.1.4.3 Building Symbols and Overlays**

Provide automated tools to assist the user in constructing new symbols and graphics overlays.

#### **10.1.4.4 Printing Preview**

When preparing graphics displays for printing, allow users to preview displays as they will appear when printed.

#### **10.1.4.5 Display Editing**

- Allow the user to add or delete symbols, labels, or other features without destroying background information.
- Allow the user to expand an area of the display as required for accurate placement of critical data.

- Provide a means for designating graphic elements for editing. Highlight selected items to provide a visual cue of forthcoming subsequent actions.
- Allow the user to reposition selected elements on the display.
- Allow the user to remove and restore selected elements.
- Allow the user to select from displays of available options when making changes to display attributes (e.g., color, symbols, line types, textures, etc.). Selection should be made by pointing rather than by naming the options.
- Provide an easy means for the user to identify attributes currently selected.
- Provide the user an easy means to change the attributes of selected graphic elements.
- Provide an easy means for naming, storing, and retrieving graphics displays and elements. Also, provide a means for reviewing and selecting from stored graphics files.

### **10.1.5 Map Display Characteristics**

#### **10.1.5.1 Map as a Base Screen**

When an application is map intensive, it is recommended that the map be used as the background or base screen, which should be the maximum display size possible to promote readability.

#### **10.1.5.2 Map Readability**

Ensure the readability of map features, since the map is the focus of the user. When possible, the screen design should avoid displays that cover the map, and windows should not obscure the map.

#### **10.1.5.3 Map Cursors**

For map cursors, use a cross-hair design that has high contrast with the background. It is recommended that cursor size subtend 20 minutes of visual angle so the average user can easily locate it on the map.

#### **10.1.5.4 Graphic Overlays**

An overlay is a layer of information (e.g., grids, boundaries, control measures) that has been drawn on a graphics canvas. Make various overlays available to the user to display (make visible), hide from display (make invisible), or delete. The preselection or filtering of graphic overlays is a recommended feature. The decluttering of graphic displays (especially maps) should be assisted.

- Carefully review labels and titles used to identify filters to ensure items are understandable. The filters should be extended to map features, such as roads, cities, vegetation, topography,

and political data. The feature overlay displays should use standard map symbols as a default (e.g., railroads, dams, and roads). The intensity of the map should be controllable to allow fadeout of the map without losing all the map features.

- Graphic overlays may overlap map features but should not obscure text information. The text may be offset with arrows to preserve map legibility.
- Include in the graphics package the capability to display a list of available overlays, distinguishing between visible and invisible overlays.
- Other possible overlays include boundary lines, oceans, rivers, grids, air fields, railways, and user-generated overlays (created through the graphics editor).
- Provide a map overlay editor function.
- Prevent the creation of overlays with the same name or title. Display the overlay feature legends at user request.

#### **10.1.5.5 Color Use with Graphic Overlays**

Using color to identify symbols is encouraged, but also ensure that it is redundant with another type of coding. This caution is especially true for friend-enemy or danger-safe designations. Dots, dashes, shapes, and video effects are recommended. Be careful to avoid visual color illusions caused by color blending (e.g., adjacent red and blue lines are seen as one purple line).

## **10.2 PRESENTATION GRAPHICS (GRAPHS, PICTURES, AND DIAGRAMS)**

Graphs should be used where necessary to visualize relationships among two or more variables, to facilitate comparing sets of data, to aid the observer in visualizing trends in data, and to aid in extrapolating future values of the underlying data. Graphs are also useful when comparing actual data to predicted values, when comparing actual values to established limits in control processes, for representing rapidly changing data, and for interpolating values between known points. In general, graphs have advantages over tabular data in summarizing complex relationships among variables and facilitate information processing and understanding.

Pictures are becoming an increasingly important form of graphic presentation. The multimedia capabilities of developing computer systems have increased the availability of pictures within computer applications. The most frequent operational picture is a map. The use of scanned maps has transferred the operational planning focus to the computer interface. The use of pictures on computer screens must be done with great care to avoid misleading the user.

Use diagrams (schematics) when the user requires information concerning the spatial relationship among objects but does not require the level of detail required by pictures. Schematic representations can be used as an aid to understanding relationships among

components of complex systems and as a means of conveying status information concerning the operation of systems and their components. They also provide a medium through which users may manipulate designs and observe subsequent actions on modeled systems.

### **10.2.1 General**

#### **10.2.1.1 Complex Formats**

Avoid complex formats, such as 3-D presentations and artistic embellishments (pictures, shading, colors, decorative items), which detract from the intended purpose of the graphic.

#### **10.2.1.2 Clarity Preservation**

Design graphics to preserve clarity when the graphics must be reproduced or reduced in size. Application window sizing should be controlled so no graphic shows partial lines.

#### **10.2.1.3 Appropriateness of Formats**

Provide formats (presentation styles) appropriate for the user's level of training and experience. Graphics should utilize user-expected symbols.

#### **10.2.1.4 Data Specific to Task**

Provide only those data the user needs for a specific task.

#### **10.2.1.5 Alternative Style Selection**

Allow users a selection of alternative presentation styles.

#### **10.2.1.6 Querying Data Elements**

Provide a means by which the user can select data elements on the graph and display the associated values.

#### **10.2.1.7 Graphical Versus Tabular**

Consider allowing the user to select between graphical and tabular data formats.

#### **10.2.1.8 Consistency**

Be consistent in design, format, labels, etc. for each presentation style.

#### **10.2.1.9 Labeling**

Clearly label the displayed graphics.

## **10.2.2 Creating And Editing**

### **10.2.2.1 Computer-Aided Entry**

Provide computer aids for the entry and organization of complex graphic data.

### **10.2.2.2 Data Entry Validation**

Validate data entries. Automated validation may include comparison to a standard range and/or the use of rules for relationships among variables. The validation process should be part of the application software.

### **10.2.2.3 Data Entry Aids to Plotting**

When plotting formats are known, provide templates or other data entry aids to facilitate the entry of graphic data.

### **10.2.2.4 Automated Plotting of Stored Data**

Automate plotting of stored data, and provide the user with automated editing and construction capabilities.

### **10.2.2.5 Automated Production of Scales**

Automate the production of scales, and/or provide the user with automated aids for scaling graphic data.

### **10.2.2.6 Lines**

- Provide automated aids for drawing straight and curvilinear line segments.
- Use rubberbanding (i.e., provide a visible line that connects a starting point to current cursor position), which can be made permanent when selected.
- Provide automated assistance in joining and intersecting line segments.
- Allow the user to identify and select line segments for moving and editing. Typically, this is done through highlighting and dragging the line. This capability should include grouping of individual segments to allow actions to be taken on the grouped object.
- Provide optional, adjustable, grid references to aid the user in aligning horizontal and vertical lines.

#### **10.2.2.7 Rule Specification by the User**

Allow the user to specify rules for attributes, relationships, and design, and have the computer apply those rules automatically during the design process. For example, straighten hand-drawn lines, adjust angles between intersecting lines, and complete details of graphic elements.

#### **10.2.2.8 Computer-Aided Drawing**

Provide computer-aided methods for drawing figures and a system of prompts or other means to aid the user in the design process.

#### **10.2.2.9 Automatic Scale Reduction**

Allow the user to edit or create drawings using a large scale, which will later automatically reduce to the desired scale.

#### **10.2.2.10 Object Manipulation**

Provide a basic set of capabilities to resize, copy, move, and rotate displayed objects. Extend these capabilities to grouped objects.

#### **10.2.2.11 Mirror Imaging**

Provide a means of producing mirror images (reflecting) as an aid in producing symmetrical graphic displays.

#### **10.2.2.12 Grouping Elements**

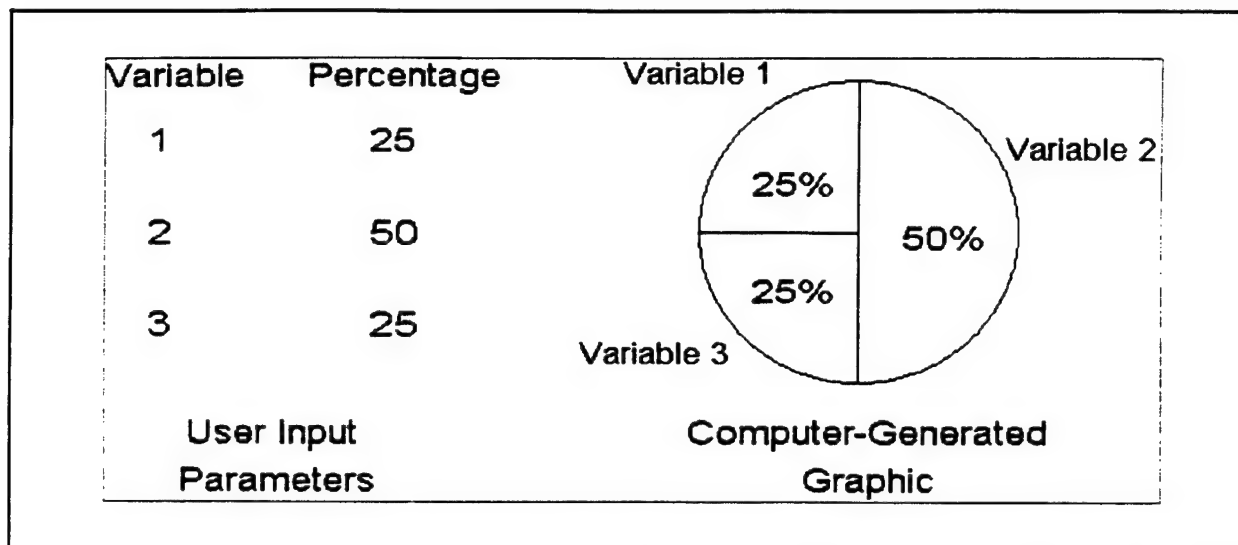
Permit the user to select and group graphic elements that will be edited in common.

#### **10.2.2.13 Area Fill Capability**

Provide an automatic means of filling enclosed areas with selected attributes (e.g., color or texture).

#### **10.2.2.14 Computer Models for Graphical Display Generation**

Provide computer models that can generate graphical displays in response to parameters provided by the user (see Figure 10-6).



**Figure 10-6. Example of How a Computer Model Can Generate Graphics From User Input**

### 10.2.3 Scales, Labels, and Coding

#### 10.2.3.1 Standard Scaling Conventions

Use standard scaling conventions: values on the horizontal axis increase to the right of the origin; values on the vertical axis increase going up from the origin. Independent variables (time or causal events) are plotted against the horizontal axis; dependent variables (effects) are plotted against the vertical axis.

#### 10.2.3.2 Standard Meanings

Use or assign standard meanings to graphic symbols, and apply them consistently.

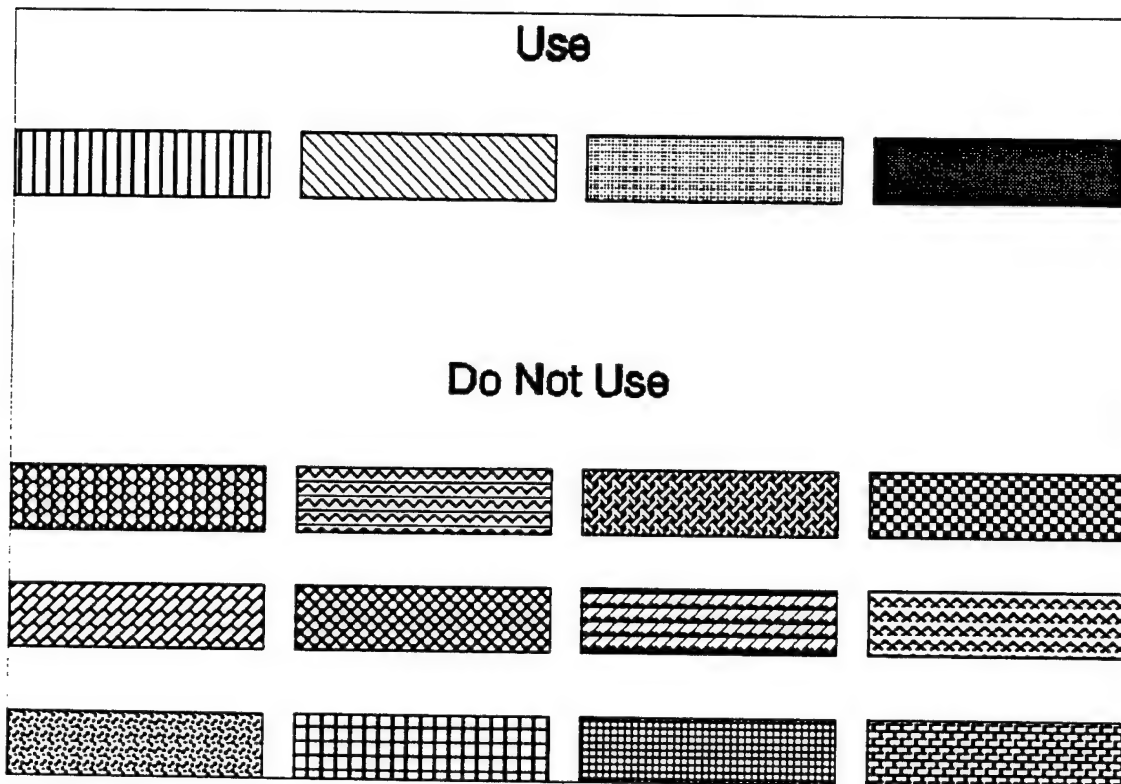
#### 10.2.3.3 Color and Pattern Coding

Users prefer colors to patterns for coding lines or filling areas of graphs on visual displays. Good design requires redundant coding be used. See Subsection 4.3 for color usage guidelines. Use texture for coding on printed outputs, since in most cases color will not be available.

#### 10.2.3.4 Texturing Displays

If texturing must be used, use simple hatching or shading, and avoid patterns that produce visual illusions of vibration and motion (see Figure 10-7).

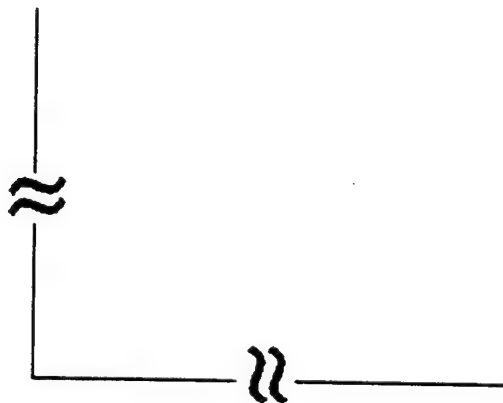




**Figure 10-7. Texture Patterns**

#### 10.2.3.5 Axes Breaks in Expanded Scales

When expanding scales to emphasize a limited range of data, provide breaks in the axes to indicate discontinuities with the origin (see Figure 10-8).



**Figure 10-8. Example of Breaks in a Graph's Axes When Scales Have Been Expanded**

#### **10.2.3.6 Duplicating Axes**

When scaled data contain extreme values, it may be difficult for the user to comprehend the scale values in relation to the data. To aid readability, add a copy of the X-axis at the top and a copy of the Y-axis at the right of the graph. Extreme values and data are thus in proximity throughout the graph. In some cases where numbers of extremely large and extremely small orders of magnitude populate the graph, a logarithmic scale may be necessary.

#### **10.2.3.7 Avoid Exaggerated Scales**

Avoid the use of exaggerated scales that distort or suppress trends in the data (see Figure 10-9).

#### **10.2.3.8 Formats for Graphic Comparison**

Provide identical formats and scales when comparisons are required between separate graphs, or plot different sets of data on the same graph.

#### **10.2.3.9 Using Linear Scales**

Linear scales should be used in preference to nonlinear scales whenever practical. For example, see Figure 10-10.

#### **10.2.3.10 Using Logarithmic Scales**

Logarithmic scales may be used where comparisons of rates of change and percentages are required or where numbers of both extremely large and extremely small orders of magnitude populate the graph.

#### **10.2.3.11 Multiple Entries**

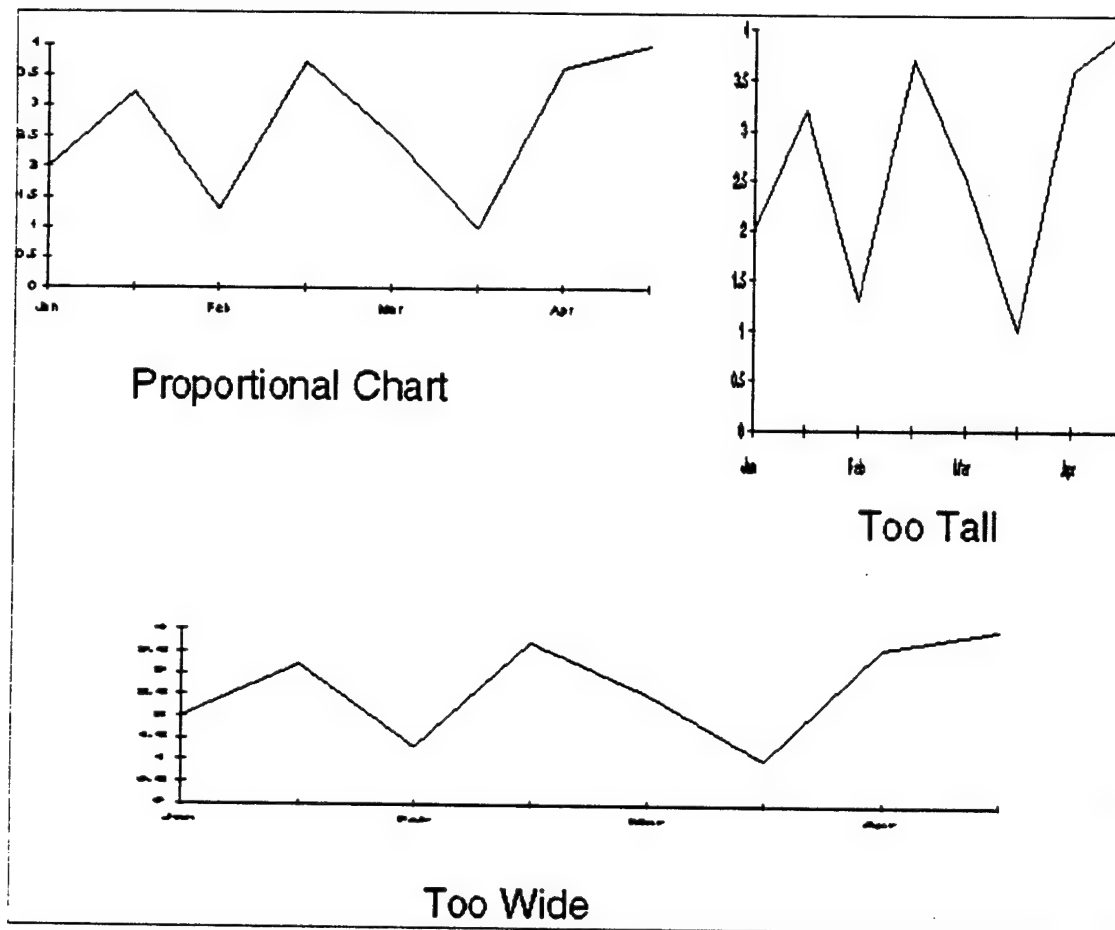
Avoid multiple scales on the axes of a single graph.

#### **10.2.3.12 Labeling Tick Marks**

Number or label tick marks corresponding to major scale divisions on the axes, and include a label containing descriptions and units of measurement on each axis.

#### **10.2.3.13 Numbering Scale Divisions**

Where practical, use no more than 10 to 12 major scale divisions separated by up to 9 subdivisions. When the appearance of the display will not be degraded, major scale divisions should be decimal multiples of whole numbers, cover the entire range of the data, and start from zero.



**Figure 10-9. Comparing Distorted Data Trends Induced by Exaggerated Scales to a Proportional Scale**

Use	2	4	6	8	10
Avoid	2	4	8	16	32

**Figure 10-10. Example of Linear Versus Nonlinear Scales**

#### **10.2.3.14 Numeric Scale Division**

Begin numeric data scales with zero when users must use displays to compare quantities or different series.

#### **10.2.3.15 Label Format**

Labels should use upper and lower case sans serif fonts and be oriented left-to-right for normal reading.

#### **10.2.3.16 Use of Labels**

Use labels in preference to legends or keys when it is necessary to identify plotted data elements. Orient labels horizontally and locate them adjacent to the referenced elements. Arrows, lines, or similar pointing conventions may also be used to connect labels to their respective data elements.

#### **10.2.3.17 Location of Legends and Keys**

Locate legends or keys that identify graphic data elements within the rectangular bounds of the graph, unless such positioning would interfere with interpretation of the displayed data.

### **10.2.4 Identifying Critical Data**

#### **10.2.4.1 Displaying Values**

Display reference or baseline values when users are required to make comparative evaluations against a fixed standard.

#### **10.2.4.2 Using Supplementary Text**

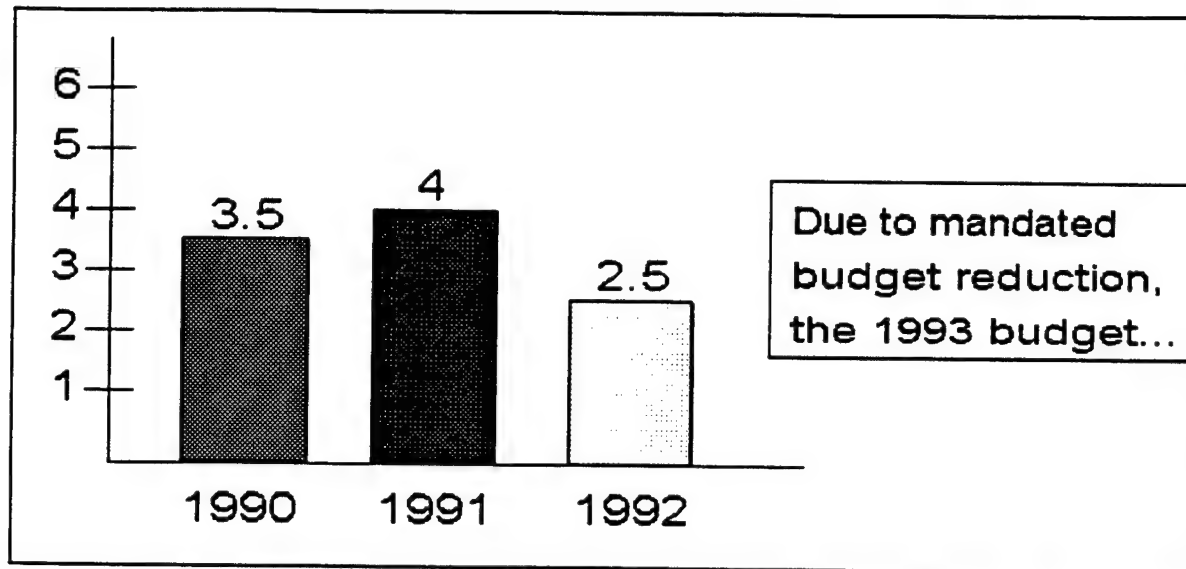
Consider using supplementary text to emphasize features of data requiring user attention. See the example in Figure 10-11.

#### **10.2.4.3 Displaying Data Values with Graphics**

Display actual data values in addition to the graphic display where precise readings of values are required, as illustrated in Figure 10-11.

#### **10.2.4.4 Position of Text Used for Labeling**

When labeling graphic data, position text consistently with respect to graphic elements.



**Figure 10-11. Use of Supplementary Text and Actual Values in a Graph**

### **10.2.5 Grid Lines**

A grid is the set of horizontal and vertical lines, including the labeled and scaled axes, which form a rectangular boundary around the graph. Additional horizontal and vertical grid lines corresponding to scale values partition the bounded area of the graph and provide a visual aid in locating and reading points on the graph(s). Use a grid and grid lines, as appropriate, when presenting data graphically.

#### **10.2.5.1 Grid Line Visibility**

Ensure that grid lines are easily distinguishable and do not obscure graphed data.

#### **10.2.5.2 Using Grid Lines**

Avoid excessive use of grid lines. Locate grid lines using the guidelines for placement of major scale values. Consider using more grid lines where greater precision is required or where the size of the display will permit their use.

#### **10.2.5.3 User Display of Grid Lines**

Where practical, allow the user to determine whether or not grid lines will be displayed.

## 10.2.6 Types Of Presentation Graphics

### 10.2.6.1 Curve and Line Graphs

- Use smoothed curves or straight lines connecting data points (line graphs) when displaying relationships between two continuous variables (e.g., when showing time variation in some quantity).
- When a single graph contains multiple curves, designate each curve with an adjacent label. If it is necessary to use a legend, list legend codes in the order in which curves occur in the graphs.
- When displaying multiple curves, highlight a curve containing critical data.
- Use line coding to distinguish among multiple curves on the same graph, and use coding consistently when the same types of data appear on different displays.
- Use a distinct line code (e.g., dashed or dotted lines) when projecting values beyond the actual data set.
- For cyclic data, provide at least one full cycle of data.
- Consider plotting the difference between two series where comparisons are necessary.

### 10.2.6.2 Area Charts

Area charts provide a means of visualizing the relative contributions of individual elements to the sum of their individual parts, often as a function of time. Figure 10-12 uses an area chart to illustrate how the total number of items (i.e., the sum of the numbers within each category) varies with time.

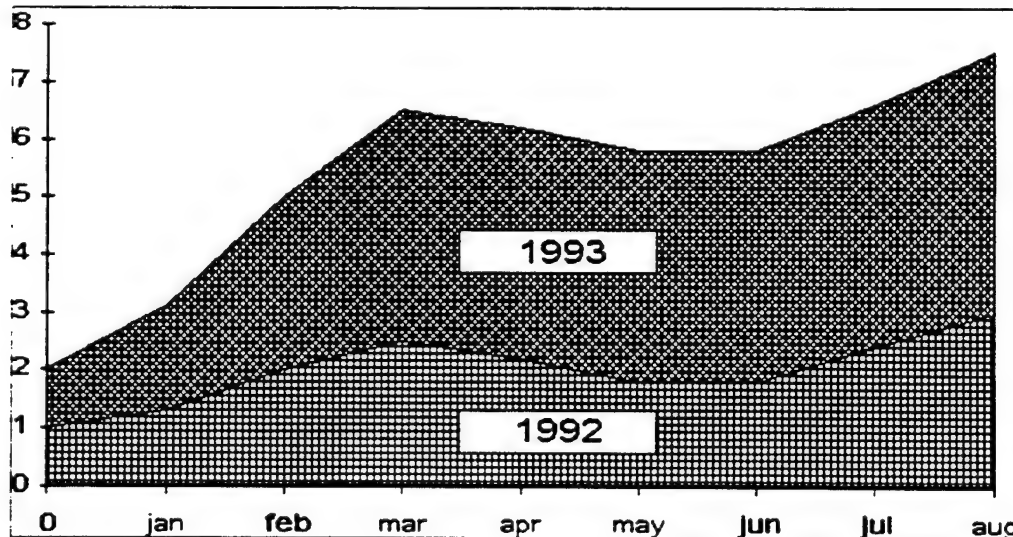


Figure 10-12. Example of an Area Chart

- Use texture or shading to indicate the area between curves.
- Stack the series with the least variable series at the bottom and the most variable at the top.
- Place labels within the textured or shaded bands if space is available.

### **10.2.6.3 Bar Graphs**

Bar graphs represent the magnitudes of numeric data by the lengths of parallel bars. Bars may be vertically or horizontally oriented and are usually spaced apart along an axis containing discrete reference points (e.g., months, mid-points of sample intervals, non-numeric categories, etc.). Histograms, or stepcharts, are bar graphs without spacing between bars, used when a large number of intervals must be plotted. Figure 10-13 illustrates bar graphs. Graphic presentations should be designed to conform to user expectations.

### **10.2.6.4 Scatterplots**

Scatterplots present data as a 2-D distribution of points and should be considered when necessary to show how variables are related or to represent the spatial distribution of data (e.g., impacts on a target). Highlight particularly significant data points. Figure 10-14 illustrates a scatterplot.

### **10.2.6.5 Pie Charts**

Pie charts, like bar graphs, are used to show proportional distribution of categories with respect to sum of the categories. See example in Figure 10-15.

- Place labels in a normal orientation on the segments of pie charts. Segment labels should include numbers that indicate percentages and/or absolute numbers represented by each segment of the display.
- Segments requiring emphasis should be highlighted or displaced slightly from the rest of the pie chart, as illustrated in Figure 10-15.

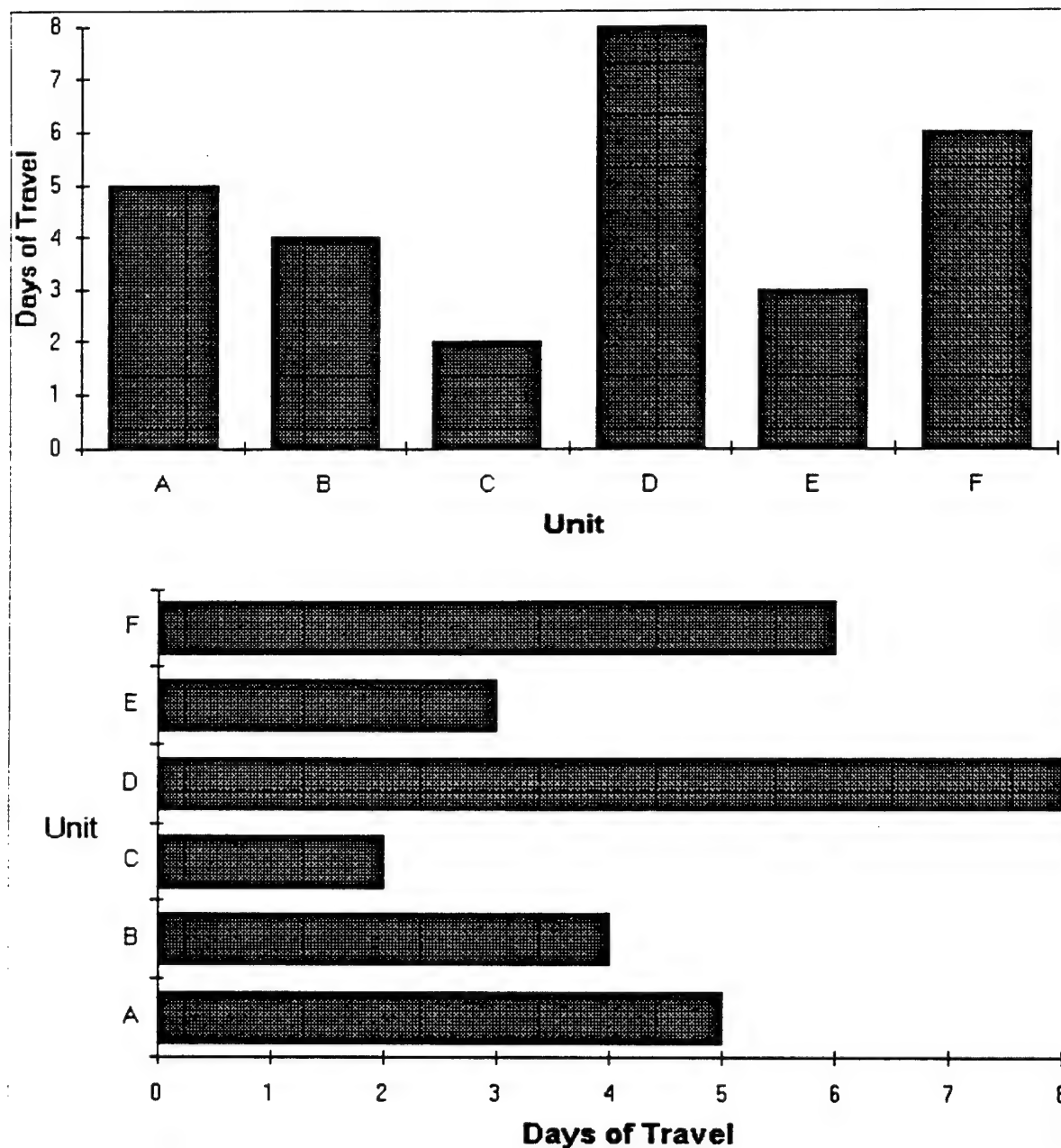
## **10.2.7 Pictures**

### **10.2.7.1 Using Pictures**

Consider using graphic pictures when a very detailed representation of objects is required. For example, see the scanned map in Figure 10-16.

### **10.2.7.2 Automated Aids for Pictures**

Provide automated aids when users must perform detailed analyses of image data.



**Figure 10-13. Examples of Bar Graphs**



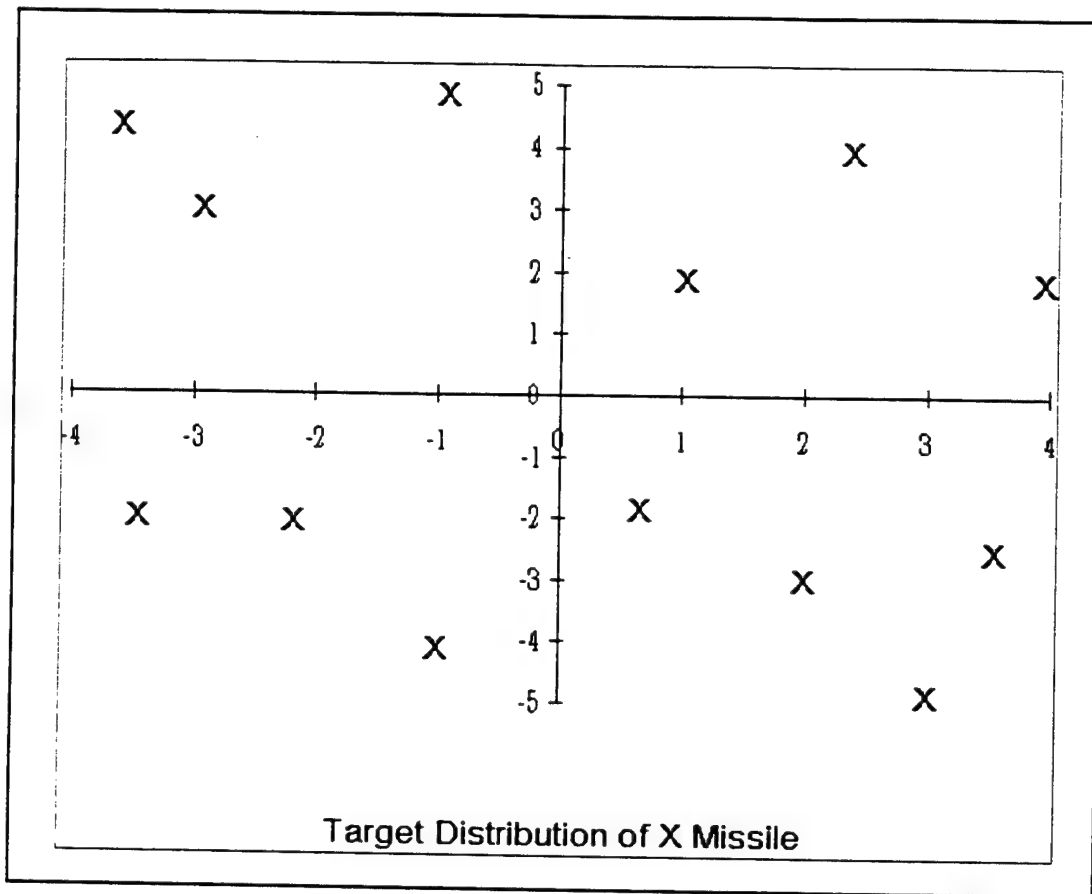


Figure 10-14. Example of a Scatterplot

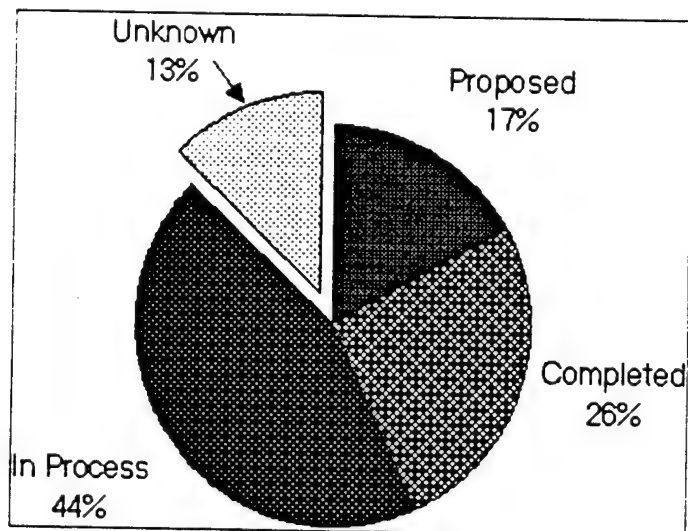
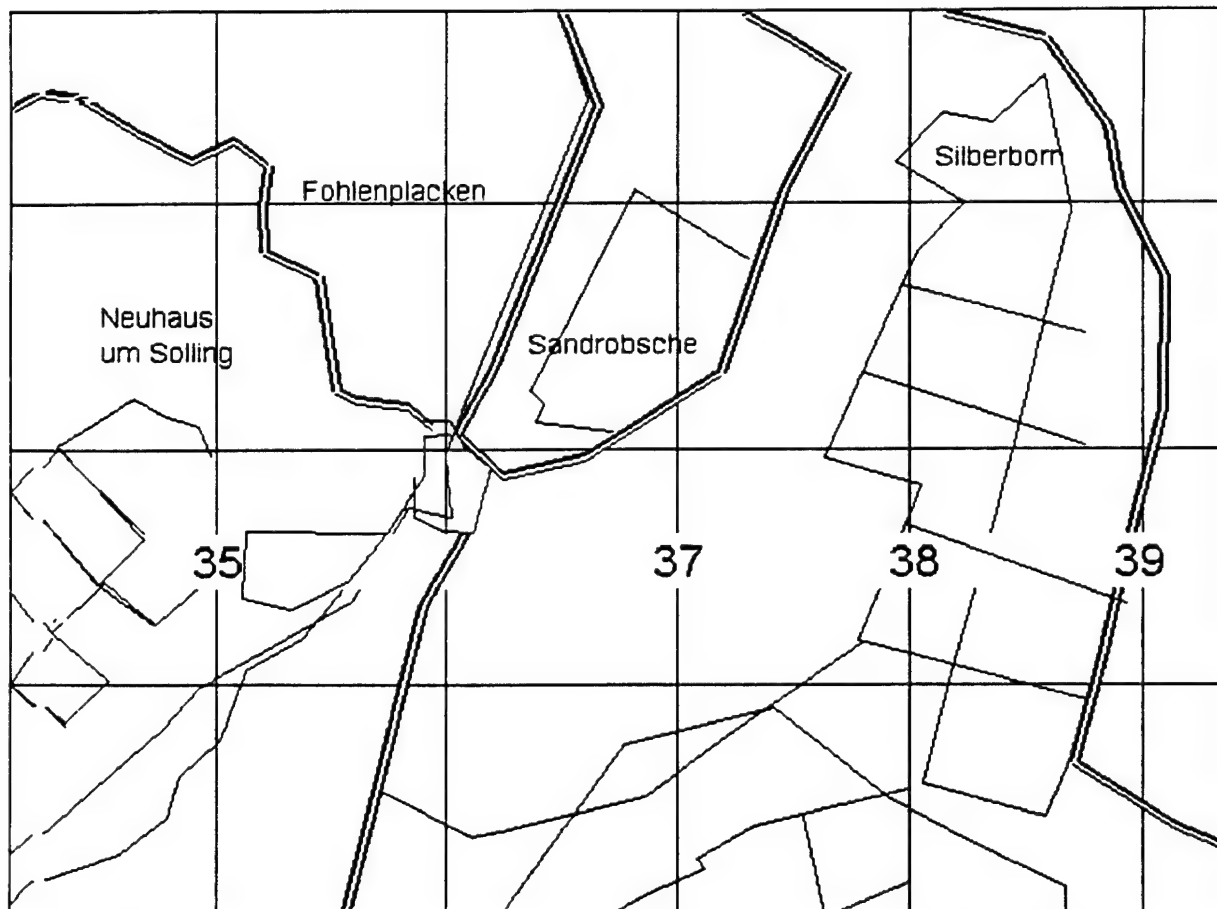


Figure 10-15. Example of a Pie Chart



**Figure 10-16. Example of a Graphic Picture**

### **10.2.8 Diagrams (Schematics)**

#### **10.2.8.1 Diagrams General**

Use diagrams when user requires information concerning the spatial relationship among objects but does not require the level of detail provided by pictures.

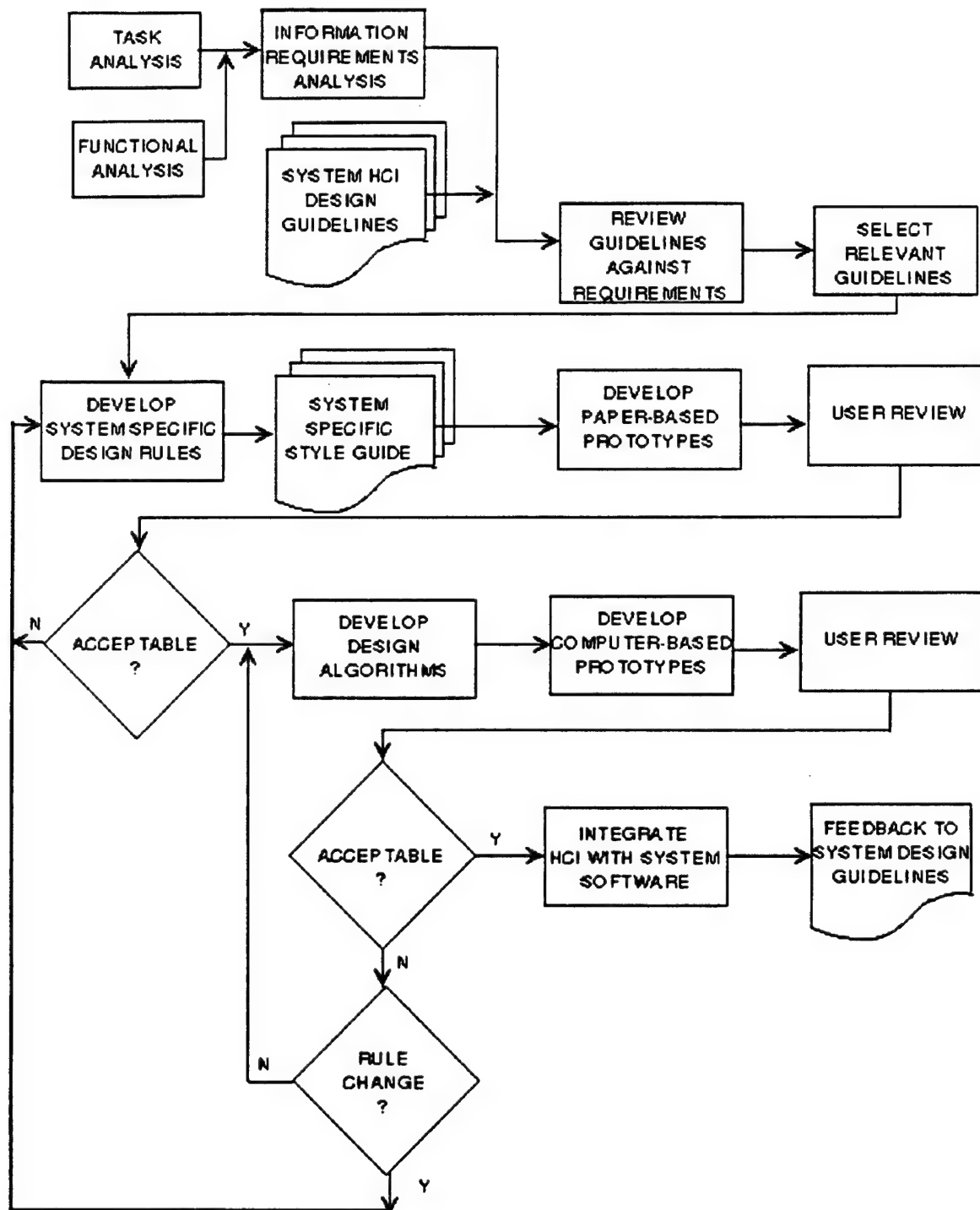
- When diagrammed data are presented in separate sections, use consistent notations across sections, provide an easy means for users to move among sections, and provide an overview of the entire diagram represented by the individual sections.
- Highlight portions of diagrams requiring special user attention.
- Provide a capability for the user to rotate displayed diagrams where it is necessary to view the object from different perspectives.

### **10.2.8.2 Flowcharts**

Use flowcharts to provide a schematic representation of sequential processes. Use them also as aids to problem-solving when solutions can be reached by answering a series of questions.

Figure 10-17 illustrates a typical flowchart.

- As appropriate, sequence flowchart elements in a logical order; otherwise, when designing flowcharts, minimize path lengths to reduce size.
- The layout of flowchart paths should conform to standard orientation conventions (i.e., left to right, top to bottom, or clockwise).
- Consistently apply the coding schemes for flowchart elements.
- Use standard directional conventions when using arrows to connect elements of flowcharts.
- Use highlighting to direct a user's attention to elements of particular significance.
- When using flowcharts as decision aids, require only one decision at each step, and provide the user with a logically ordered list of available options.
- Use consistent wording for options displayed at decision points.



**Figure 10-17. Sample Flowchart**

## REFERENCES

Paragraphs	References
10.1.1.1	Smith and Mosier (1986) para 2.4.8-3
10.1.1.2	Smith and Mosier (1986) para 2.4.8-3
10.1.1.3	Smith and Mosier (1986) para 2.4.8-15
10.1.1.4	Smith and Mosier (1986) para 2.4.8-4 and 2.4.8-5
10.1.1.5	Smith and Mosier (1986) para 2.4.8-7 through 2.4.8-9
10.1.1.6	Smith and Mosier (1986) para 2.4.8-13
10.1.2.2a	Lewis and Fallesen (1989) para 2.5.6.1.1.2
10.1.2.2b	Smith and Mosier (1986) para 1.6-17
10.1.2.3	U.S. Department of the Army (1985b)
10.1.2.3a	Lewis and Fallesen (1989) para 2.5.6.1.1.1; U.S. Department of the Army (1985b)
10.1.2.3d	Lewis and Fallesen (1989) para 2.5.6.1.1.3
10.1.2.3e	Lewis and Fallesen (1989) para 2.5.6.1.1.4
10.1.2.3f	Smith and Mosier (1986) para 2.6-8
10.1.2.3g	Lewis and Fallesen (1989) para 2.5.6.1.1.4.6; U.S. Department of the Army (1985b)
10.1.2.3h	Lewis and Fallesen (1989) para 2.5.6.1.3
10.1.2.4	Bowser (1991) p. 13
10.1.2.4a	Lewis and Fallesen (1989) para 2.5.6.1.1.2 and 2.5.6.2.5
10.1.2.4c	Smith and Mosier (1986) para 2.4.8-12
10.1.3	Bowser (1991) p. 13
10.1.3.1a	Lewis and Fallesen (1989) para 2.5.6.2.3
10.1.3.1b	Smith and Mosier (1986) para 2.4.8-11
10.1.3.2e	Lewis and Fallesen (1989) para 2.5.6.2.4
10.1.3.2f	Smith and Mosier (1986) para 2.4-16
10.1.3.2g	Smith and Mosier (1986) para 2.4-17
10.1.3.3b	Smith and Mosier (1986) para 2.4.8-18

## REFERENCES (cont'd)

### Paragraphs

### References

10.1.3.3e	Smith and Mosier (1986) para 2.7.3-4
10.1.3.4e	Smith and Mosier (1986) para 2.4-18
10.1.3.5	Bowser (1991) p. 14
10.1.3.6	Bowser (1991) p. 14
10.1.4.2	Bowser (1991) p. 15
10.1.4.3	Smith and Mosier (1986) para 2.4-20
10.1.4.5b	Smith and Mosier (1986) para 1.6-5
10.1.4.5c	Smith and Mosier (1986) para 1.6-6 and 7
10.1.4.5e	Smith and Mosier (1986) para 1.6-9
10.1.4.5f	Smith and Mosier (1986) para 1.6.10 and 11
10.1.4.5g	Smith and Mosier (1986) para 1.6-12
10.1.4.5h	Smith and Mosier (1986) para 1.6-13
10.1.4.5i	Smith and Mosier (1986) para 1.6-15 and 16
10.1.5.1	Bowser (1991) p. 14
10.1.5.2	Bowser (1991) p. 14
10.1.5.3	HFS (1988); Bowser (1991) p. 14
10.1.5.4	Bowser (1991) p. 14
10.1.5.4b	Bowser (1991) p. 15
10.1.5.5	Bowser (1991) p. 15
10.2	Smith and Mosier (1986) para 2.4.6-2, 5.2
10.2.1	Lewis and Fallesen (1989) para 2.1.1-3; Brown (1989) para 5.3-8, 5.10; Brown (1989) p. 88
10.2.1.1	Lewis and Fallesen (1989) para 2.3.6
10.2.1.2	Bowser (1991) p. 15; Lewis and Fallesen (1989) para 2.3.9
10.2.1.4	Smith and Mosier (1986) para 2.4-5
10.2.1.6	Lewis and Fallesen (1989) para 2.3.2.2b
10.2.1.7	Lewis and Fallesen (1989) para 2.3.2.2c

## REFERENCES (cont'd)

### Paragraphs

### References

10.2.1.8	Smith and Mosier (1986) para 2.4-4
10.2.2.1	Smith and Mosier (1986) paras 1.6.1-1 and 1.6.1-18
10.2.2.2	Smith and Mosier (1986) para 1.6-19
10.2.2.3	Smith and Mosier (1986) para 1.6.1-2
10.2.2.4	Smith and Mosier (1986) paras 1.6.1-2 and 1.6.1-4
10.2.2.5	Smith and Mosier (1986) paras 1.6.1-5 and 1.6.1-6
10.2.2.6a	Smith and Mosier (1986) para 1.6.2-1
10.2.2.6b	Smith and Mosier (1986) para 1.6.2-2
10.2.2.6c	Smith and Mosier (1986) para 1.6.2-3
10.2.2.6d	Smith and Mosier (1986) paras 1.6.2-4 and 1.6.2-5
10.2.2.7	Smith and Mosier (1986) paras 1.6.2-6, 1.6.2-7 and 1.6.2-18
10.2.2.8	Smith and Mosier (1986) paras 1.6.2-8 and 1.6.2-9
10.2.2.9	Smith and Mosier (1986) para 1.6.2-11
10.2.2.10	Smith and Mosier (1986) paras 1.6.2-10, 1.6.2-12 and 1.6.2-13
10.2.2.11	Smith and Mosier (1986) para 1.6.2-14
10.2.2.12	Smith and Mosier (1986) para 1.6.2-15
10.2.2.13	Smith and Mosier (1986) para 1.6.2-17
10.2.2.14	Smith and Mosier (1986) para 1.6.2-19
10.2.3.1	Smith and Mosier (1986) para 2.4.1-1
10.2.3.2	Smith and Mosier (1986) para 2.4-12
10.2.3.3	Brown (1989) para 4.2.7
10.2.3.4	Smith and Mosier (1986) para 2.4-14; Lewis and Fallesen (1989) para 2.4.7.1
10.2.3.5	Smith and Mosier (1986) para 2.4.1-7
10.2.3.6	Smith and Mosier (1986) para 2.4.18
10.2.3.9	Lewis and Fallesen (1989) para 2.3.1.3
10.2.3.10	Lewis and Fallesen (1989) para 2.3.1.4

## REFERENCES (cont'd)

### Paragraphs

### References

10.2.3.11	Lewis and Fallesen (1989) para 2.3.1.5
10.2.3.12	DoD (1989a) para 5.15.3.6.4
10.2.3.13	Lewis and Fallesen (1989) para 2.3.1.8-9
10.2.3.14	Smith and Mosier (1986) para 2.4.1-7
10.2.3.15	Lewis and Fallesen (1989) para 2.3.3
10.2.3.16	Lewis and Fallesen (1989) para 2.4-5
10.2.3.17	Lewis and Fallesen (1989) para 2.3.5
10.2.4.1	Smith and Mosier (1986) para 2.46
10.2.4.2	Smith (1986) para 2.4-7
10.2.4.3	Smith and Mosier (1986) para 2.4-8
10.2.4.4	Smith and Mosier (1986) para 2.4-9
10.2.5	Lewis and Fallesen (1989) para 2.3.2
10.2.5.1	Lewis and Fallesen (1989) para 2.3.2.1
10.2.5.2	Lewis and Fallesen (1989) para 2.3.2.1.1
10.2.5.3	Lewis and Fallesen (1989) para 2.3.2.2a
10.2.6.1a	Smith and Mosier (1986) para 2.4.3-1
10.2.6.1b	Smith and Mosier (1986) para 2.4.3-3, 2.4.3-4
10.2.6.1c	Smith and Mosier (1986) para 2.4.3-5
10.2.6.1d	Smith and Mosier (1986) paras 2.4.3-6 and 2.4.3-7
10.2.6.1e	Smith and Mosier (1986) para 2.4.3-8
10.2.6.1f	Smith and Mosier (1986) para 2.4.3-10
10.2.6.1g	Smith and Mosier (1986) para 2.4.3-11
10.2.6.2a	Smith and Mosier (1986) para 2.4.3-12
10.2.6.2b	Smith and Mosier (1986) para 2.4.3-13
10.2.6.2c	Smith and Mosier (1986) para 2.4.3-14
10.2.6.3	Bowser (1991) p. 15; Smith and Mosier (1986) para 2.4.4-3



## REFERENCES (cont'd)

### Paragraphs

### References

10.2.6.4	Smith and Mosier (1986) para 2.4.2-3
10.2.6.5	Smith and Mosier (1986) para 2.4.5-1
10.2.6.5a	Smith and Mosier (1986) paras 2.4.5-2 and 2.4.5-3
10.2.6.5b	Smith and Mosier (1986) para 2.4.5-5
10.2.7	Smith and Mosier (1986) para 2.4.6-1
10.2.7.1	Smith and Mosier (1986) para 2.4.6-6
10.2.8.1a	Smith and Mosier (1986) para 2.4.6-3
10.2.8.1b	Smith and Mosier (1986) para 2.4.6-3
10.2.8.1c	Smith and Mosier (1986) para 2.4.6-5
10.2.8.2	Smith and Mosier (1986) paras 2.4.7-1 and 2.4.7-2
10.2.8.2a	Smith and Mosier (1986) paras 2.4.7-3 and 2.4.7-4
10.2.8.2b	Smith and Mosier (1986) para 2.4.7.5
10.2.8.2c	Smith and Mosier (1986) para 2.4.7-6
10.2.8.2d	Bowser (1991) p. 16; Smith and Mosier (1986) para 2.4.7-7
10.2.8.2e	Smith and Mosier (1986) para 2.4.7-8
10.2.8.2f	Smith and Mosier (1986) paras 2.4.7-9 and 2.4.7-10
10.2.8.2g	Smith and Mosier (1986) para 2.4.7-12

**This page intentionally left blank.**

## 11.0 DECISION AIDS

DoD continues to develop automated decision aids in support of user tasks. Although what constitutes a decision aid has been debated, it is important to point out that decision aids assist, rather than replace, human decision-makers. Consequently, when defining decision aids, applications limited to managing information are usually excluded, as are those that make decisions in a fully autonomous mode.

The question of "when to use decision aids" is reviewed in the first part of this section. Given that decision aids are to be used, the next step is to define the requirements. When the requirements are firm, the features needed to support the requirements become important. Section 11.0 then deals with specific issues of decision aid interface design.

Decision aids may be designed to be parts of other software or as stand-alone applications. For example, decision aids have been designed to assist users in evaluating military courses of action. These applications present alternatives and supporting evidence, as well as assist the user in evaluating the alternatives. The user retains a major role in developing the final recommendations. Information management software such as database management, text processing, and graphics applications may support the decision process but are not usually considered decision aids. Other applications, such as engine diagnostic software, may include many relatively fixed rules derived from human experts. Such "expert systems" can include many properties of decision aids, but they place relatively more emphasis on internal rules to arrive at conclusions. Examples of autonomous systems include automatic fire control systems and robotic devices. These systems may require human supervision but rely heavily on internal rules and algorithms for their operation.

It is difficult to make a distinction between decision aids and expert systems (which include autonomous capabilities), since both require cooperation between human and automated system components. Holtzman (1989) differentiates between expert and intelligent decision systems and points out which is appropriate to use based on subject matter, circumstance, and preference of the decision-maker. Expert systems may have a relatively large knowledge base and rules that respond to constant environmental factors, whereas decision aids place more burden on the decision-maker. Decision aids provide assistance and are designed to help in uncertain or novel situations. The guidelines presented in this section are appropriate for both expert systems and decision aids. Therefore, the term "decision aid" or "aid" will be used to refer to both types of decision support applications.

### 11.1 USE OF DECISION AIDS

Decision aids should be used to compensate for known limitations in human decision-making and to offset the adverse effects of external factors. In general, difficulties can arise because of the fundamental limits of human cognitive (i.e., mental) abilities and lack of experience. Difficulties also arise because of various environmental factors that both stress the decision-maker and determine the type, quantity, quality, and rate of information presented.

### 11.1.1 Cognitive Considerations

Consider the cognitive limitations and styles of decision-makers when designing decision aids. For example, overload (i.e., stress, information, situational, etc.) often causes users to focus on a subset of the available information. Innate abilities and learned information-processing strategies may cause additional problems. The following points describe several commonly occurring limitations.

- **Cognitive Limitations** - Novices, individuals lacking confidence, and those performing tasks under stressful conditions will make errors that may result in less-than-optimal decisions. For example:
  - Humans often have difficulty retrieving, retaining, representing, and manipulating large amounts of information. They also may have difficulty combining multiple cues or criteria or performing computational tasks. These difficulties result in delaying performance or avoiding difficult tasks.
  - Humans often have difficulty making decisions in times of uncertainty.
  - Novices do not have previous experience, so they often fail to recognize errors.
  - When making a decision, humans often simplify decision problems by selectively perceiving data, information, and knowledge. They may set outcome objectives and then look for a decision that meets them. Thus, they may focus on confirming rather than on refuting evidence. They also may adjust decision methods to fit goals or desired results.
  - If a decision leads to a negative result, humans may attribute the outcome to chance or to the complexity of the problem, rather than to their own decision-making deficiencies.
  - Humans have limited memory available for current tasks and will lose some information within seconds.
  - Humans have limited abilities to organize information.
  - Humans usually have difficulties with symbolic and quantitative manipulation of mental representations, and they may have difficulty formulating or dealing with abstractions.
  - Humans may have difficulty extrapolating time and space information.
  - Humans may fail to use prior experience to generalize in new situations.
- **Pitfalls of Complexity** - Humans often have difficulty dealing with complexity and may, therefore, try to make a problem less complex by avoiding certain aspects of it. Consequently, they may not consider all factors when making decisions. Some strategies humans use to make decisions less complex are:

- Humans often simplify decision problems by only considering a few alternatives.
- Humans may use only part of the available information. Or they may use information that corresponds to a mental representation or model of what they imagine the solution to be, even if this means rejecting or misperceiving relevant information. They may combine or “chunk” information in various ways, rely on poor memory-search strategies, or rely on erroneously perceived correlations between data.
- **Cognitive Biases** - Humans have biases that can carry over into the decision-making process. Humans may:
  - Recall information that has been recently acquired, frequently rehearsed, or semantically related to current information
  - Anchor their judgments (i.e., place greater emphasis on early evidence) and then fail to adjust when provided new information
  - Give preference to information they believe is causally related to the problem
  - Provide numeric judgments that contain systematic bias or variance
  - Select cues that are often unreliable indicators of the true situation
  - Use inappropriate analogies to generate and compare options
  - Incorrectly identify current situations with similar past events
  - Fail to detect unique features among similar cases, and inconsistent or ambiguous information may not be noticed or emphasized appropriately.
- **Time Allocation** - Humans may fail to allocate time properly to different phases of the planning process. Too much attention to early stages of planning may leave inadequate time to evaluate derived alternatives properly. Humans may:
  - Perform detailed analysis early but fail to do so later
  - Fail to develop and evaluate the alternatives thoroughly
  - Fail to identify, evaluate, compare, and combine salient information and, therefore, fail to identify, prioritize, and assess goals
  - Fail to model (war-game) alternatives because of lack of time.

#### 11.1.2 External Factors

Many external factors may influence the quality of decisions.

- **Information overload** - When the complexity, dynamics, and/or volume of information to assess are high (such as in battlefield operations), they may degrade decision-making performance.
- **Time stress** - Humans have difficulty analyzing information fast enough to meet external time constraints. When they have enough time, they often have difficulty maintaining high performance long enough to analyze all data.
- **Limited information** - Decision-makers may work in situations where information available to support their decisions is limited. Under such circumstances, lack of experience and human limitations in making or formulating estimates may pose problems.
- **Training** - Decision-makers may not have the experience, deductive skills, or knowledge of the procedures necessary to make a decision. A decision aid can assist by performing some steps, by leading the human through the required steps, and by filling in knowledge gaps. Properly designed decision aids also train users through explanation and embedded training.

### **11.1.3 When to Use Decision Aids**

Use decision aids to help the user overcome the difficulties previously described. The following are examples.

#### **11.1.3.1 Manage Complexity**

Decision aids help the user cope with information overload; they focus attention. Use decision aids when the user is trying to manipulate large amounts of data or visual representations, combining multiple criteria, allocating resources, managing detailed information, and selecting and deciding among alternatives.

#### **11.1.3.2 Improve Timeliness**

A decision aid helps a user perform many time-consuming activities more quickly. Some examples include diagnosing the current state of a system and mathematical calculations, particularly when they are beyond the user's abilities. Providing aid when users encounter unfamiliar problems also helps improve the timeliness of the process.

#### **11.1.3.3 Best Use of Limited Data**

Use decision aids when limited data result in uncertainty. Decision aids help by predicting future events from limited information, improving the accuracy and reliability of critical tasks, and addressing critical areas beyond the ability of the user.

#### **11.1.3.4 Overcoming Limitations**

Use decision aids to overcome the human cognitive limitations described in the earlier sections. For example:

- Use decision aids to overcome human limitations in dealing with uncertainty.
- Decision aids are helpful in overcoming emotional components of decision-making.
- If the quality of human performance is in question, decision aids can add greater accuracy to the process.
- Decision aids are ideal in cases where memory- and information-retention problems exist.
- Decision aids overcome cognitive biases well.

#### **11.1.4 When to Consider Alternatives**

The following are circumstances under which the use of decision aids may not be advisable.

##### **11.1.4.1 Obvious Solutions**

Do not use decision aids when solutions are obvious or when one alternative clearly dominates all other options.

##### **11.1.4.2 Time Requirements**

Use decision aids only when sufficient time is available or when the user is authorized to make decisions.

##### **11.1.4.3 Generalizing**

As appropriate, defer to the human ability to generalize.

##### **11.1.4.4 Adaptation**

Recognize situations where individuals may be superior in adapting to novel situations.

#### **11.1.5 Cautions and Limitations**

Exercise caution when introducing decision aids, in particular, when they include functions that reduce the role of human judgment.

##### **11.1.5.1 User Complacency**

The decision aid may encourage users to take a less active role. This, in turn, may cause users to be inattentive and less prepared to handle sudden decreases or increases in workload, both of which may reduce accuracy.

#### **11.1.5.2 Continued Vigilance**

If the user's role becomes less active, the user may have difficulty maintaining sustained attention, which may lead to longer user response times.

#### **11.1.5.3 Discrimination Limitations**

It is necessary to recognize limitations in a user's ability to discriminate between correct and incorrect automated decisions.

#### **11.1.5.4 Fear of Automation**

Many users mistrust automation and automated decisions, preferring to believe their performance is superior to that of automated systems. User attitudes toward automation are often based on the fear of being replaced. The designer should take this into account when planning the role and degree of authority the user will have in overriding automated decisions.

### **11.2 DEFINING DECISION AID REQUIREMENTS**

Develop decision aids or expert systems that focus on tasks that users find difficult, rather than on what is already done routinely.

#### **11.2.1 Understand Tasks**

Base designs on an in-depth understanding of both the tasks to be performed and the conditions of their performance.

- The best way to define decision aid requirements is to start with experts in the field. However, it is important to choose the experts appropriately. Be sure to use more than one expert and verify that they really are experts. If they have knowledge of part of the field, be sure to consider those parts, and find other experts for the other parts. Also, ensure that common users participate with the group of experts. When obtaining information from the experts, provide a means to identify the criteria used to reach decisions.
- Decision aids must be matched to the situation and limitations they are designed to support.
- Recognize that not all functions are appropriate for decision aids. Determine the appropriate functions and design them to be compatible with the user's decision processes.
- Provide no more than one aid for each task.

#### **11.2.2 Understand Requirements**

Decision aid development should be driven by requirements, not by technology.



- Identify areas where users actually need help, then match the decision aid to the needs of the intended users.
- Recognize the user's decision situation and goals, and focus on the highest-level goal.
- Anticipate skepticism concerning automated decision support. Recognize that the dominant factor in accepting decision aids is perceived utility. The system must add new capabilities or increase efficiency in the performance of decision-making tasks.
- Consider characteristics of the user population in designing the decision aid and its interface.

### **11.2.3 Types of Aids**

Types of aids and presentation formats may vary according to the phases of the decision process (i.e., alerting, acquisition, evaluation, and responding) and factors such as time stress.

### **11.2.4 Function Allocation Between Humans and Computers**

Allocating functions between humans and computers must be based on cognitive task analysis, not on what is achievable using current technology.

- Recognize that aided performance may not exceed unaided performance, even though aided methods are preferred.
- Decision aids and expert systems can enhance decision quality. However, they may increase the user's workload because users may be required to consider more variables. Seek design alternatives that prevent or minimize increased workload.
- The aid must be complete for its intended purpose. Address all critical aspects of the decision situation.
- Recognize that a user's decision-making behavior is contingent upon the task and context within which it is performed. Design the decision aid to provide decision methods suitable to probable variations in tasks and context.
- Users often prefer to perform some of the tasks and allow the decision aids to perform others. Specifically, users prefer to do the easy to moderately difficult tasks and leave difficult tasks to the decision aid. This interaction is necessary to maintain user interest and attention. Decision aids are more acceptable to users if viewed as advisors rather than decision-makers.
- Avoid applications that are trivial or lack complexity because they may undermine the value of automated decision support methods.

## **11.3 FEATURES OF DECISION AIDS**

### **11.3.1 General Design Considerations**

- Ensure that a decision aid is easier to use than the decision process it replaces. It must be flexible, versatile, and easy enough to benefit typical users (i.e., users don't need to be subject matter experts). A decision aid must use terminology and criteria appropriate to the target user group. It must be easy to control and understand.
- Ensure that a decision aid is capable of responding to the user's ad hoc requests in time to allow the information to influence decisions. The interface should facilitate the exchange of information.
- Tailor decision aids to the resources available to the user.
- Ensure that a decision aid automatically identifies meaningful patterns and relationships and brings them to the attention of the user.

### **11.3.2 Provide Decision Alternatives**

- Ensure that a decision aid is able to support development and evaluation of multiple, feasible alternatives. The aid should present a set of possible alternatives, each of which could be feasible. However, the aid should not display all of the options when that would be too complex. The decision aid also should display which goals are served by the different alternatives and applicable options.
- Ensure that the decision aid supports user evaluation of decision options. First, the aid should generate alternatives for the user to evaluate and should allow the user to input his or her own alternative(s). Second, the aid should have a method of assigning and explaining probabilities for alternatives. The user should be able to explore different solutions, including using different decision strategies and criteria. Once the user has applied all desired options, the aid should rank-order the decision alternatives. This assistance also should include guidance in using rating procedures.

### **11.3.3 Prediction, Simulation, and Modeling**

- Ensure that the application is able to predict future data. Historical data should be available to make comparisons, search for precedents, and assist the user in visualizing trends. The decision aid should alert the user when it predicts a future problem or opportunity upon which the user needs to act.
- Provide a modeling and simulation capability to support "what if?" exercises and to make predictions based on current conditions.
- Ensure that models used in decision aiding are appropriate, designed to answer specific questions, and validated.

#### **11.3.4 Identify and Assess Factors Underlying Decisions**

- Provide a means of obtaining and assessing weights for multiple criteria. Multiple criteria should be statistically independent, when possible. As appropriate, provide a means of combining weights from multiple sources. This refers to the technique of multi-attribute decision-making.
- Identify and rank causal factors by their importance, and assign weights. The application should allow users to modify the decision factors and their weights and to provide and adjust risk factors used in decision models. This refers to techniques such as pair-wise comparison.
- Ensure that the aid is able to explain the contributions of underlying factors and supports the use of sensitivity analysis for exploring those contributions. The aid must identify and assess operational constraints and provide a means of informing the user (upon request) of decision aid boundaries or other limitations. The aid should make available to the user the assumptions underlying modes and parameters and a history of the aid's past performance.
- Ensure that the decision aid makes it easy for the user to provide input into the aid's decision. The user should be able to add new decision factors and set the range of conditions (within the decision aid's set limits), the level of output detail, and the parameters for optimization. Provide a means for saving and reusing the user's modifications, but also provide a means to return to the default settings.
- Assist in visualizing interacting factors.
- Provide a means for assuring the validity of elements added to the decision model, in particular those used over successive applications.

#### **11.3.5 Handling Decision Aid Recommendations**

- Ensure that the application is able to calculate and display results of selected decision options.
- Ensure that the application provides facilities for assessing costs, risks, and benefits of all alternatives.
- For users to trust the decision aid, the aid must explain the rationale behind outputs or recommendations. The aid also should provide indicators of certainty or uncertainty when making recommendations.
- When data are missing or uncertain, ensure that the aid identifies this situation and gives information on the possible impact on the recommendations.
- Ensure that the decision aid includes internal consistency checks to prevent the system from making contradictory predictions and recommendations.
- Ensure that the decision aid informs the user when it cannot handle the current situation.

## 11.4 USER REQUIREMENTS

### 11.4.1 General Considerations

- Ensure that the decision aid is user-friendly, beneficial to the user, and presents information that is readily understood by or familiar to the users. Where possible and appropriate, ensure that the decision aid has sufficient “intelligence” to adjust to user task requirements.
- Ensure that the decision aid uses decision methods acceptable to the decision-maker and is able to accommodate user changes. Once the decision method is determined, the user must retain control throughout the process. The aid should provide feedback on the method and the current stage of processing.
- Reduce the user’s data-entry requirements as much as possible. To do so, set defaults for data-entry fields. However, these defaults and fields must be user-changeable.
- Ensure that a decision aid automatically alerts users to important new developments occurring in the database or as a result of predictive modeling.
- Ensure that the system encourages the user to participate in the decision process. To do this, the system should represent problems and solutions in the same way the users do. The system also should try to foster user “ownership” of decisions and allow the user to exercise judgment over the decision aid results. This includes providing sufficient information to the user both about the process and about the end result.
- Ensure that the decision aid guides the user through the process, providing automated guidance on how to define and analyze a problem and formulate a decision. When user input is required, the decision aid should help make this requirement clear. However, it should not make the user dependent, such that the process cannot be completed when the system is unavailable.
- Avoid presenting too much data. Use aids to reduce, filter, and preprocess data into a form useful to the decision-maker.
- Avoid increasing the user’s work load, when possible. Prepare users for changes and possible increases in work effort when necessary, and point out the aid’s abilities to increase effectiveness.
- Reduce complexity. A major reason for using decision aids is to simplify the user’s task. Therefore, some guidelines are necessary on the amount of information to present to the user. In general, the system should provide information required to perform the tasks allocated to the user; however, it should only present information relevant to the task being performed. The system should provide no more information than is essential and should avoid repeating already available information. Present the information using a level of abstraction, resolution, or detail appropriate to the immediate task.

- When time is limited, ensure that the system anticipates the user's needs and provides a greater degree of autonomous decision-making.
- Ensure that users are able to extend and personalize the decision aid. However, provide a means to validate models created or modified by users, and provide sufficient warnings about the consequences of failures to validate. Ensure that the decision aid can be easily returned to a default state.
- Ensure that the decision aid analysis is flexible to the user's needs and desires; give the user control over the data retrieval and analysis process. The user should select the degree of analysis to be done and time frame to be considered. When the system asks questions, the user should have the option of either changing the question or not answering. The system should also accommodate the various information requirements of commanders and staff users, including the ability to adjust the level of detail. It should be able to create a user profile containing preferences and jargon.
- Provide procedures appropriate to the user's level of expertise. Designers should recognize that experts may use mental imagery; novices depend more on rule-based procedures.

#### **11.4.2 Decision Aid Interface**

- Ensure that the user-machine interface supports an intelligent dialogue between the user and the decision aid. For example, it should adapt to the user; understand the user's goals, needs, and abilities; interpret poorly formulated queries; correct user errors; and overcome user limitations. The interface also should reflect the tasks to be performed and should be tailored to the resources available.
- Ensure that the system helps prevent the user from making errors. When errors are made, it should provide automatic error recovery.
- Apply user-interface design guidelines mentioned elsewhere in this document.
- Ensure that the decision aid allows users to customize formats to their own needs. However, it is preferable to minimize the user's requirements to make such changes. To do this, the application should associate and group data in a meaningful way, and displays should match the task.

See Subsection 8.2 of this *Style Guide* for detailed information on HELP applications.

### **11.4.3 Explanations**

Decision aids should be capable of providing domain-specific explanations to answer user questions and must be capable of guiding the user through the decision process, as well as providing procedural help on system use.

- When the system provides explanations, ensure they are easy for the user to understand. Explanations should use terms familiar to the user, incorporating the user's concept of the problem and maintaining consistency with the immediate task. Intuitive explanations or analogies are helpful for topics that are likely to be too difficult for the user to understand.
- Length of explanation is important. Provide a short explanation initially, with the ability to provide more detail at the user's request. Consider how much to tell the user. Weigh trade-offs in what the user can learn about the decision aid and what the decision aid can/should explain to the user.
- Assist the user in locating key elements of the decision model, as related to a specific decision task.
- Provide the capability to explain the current decision model or method, and be prepared to justify the use of component factors. Document the decision aid's algorithms, and make them available for user inspection.

### **11.4.4 Training**

- Provide backup systems and appropriate training in performing any user tasks replaced by decision aids. When decision aids are available, provide regular training to the user in all skills required to maintain proficiency on backup systems. This training will be necessary if decision aids become unavailable. Training may be preferable to using decision aids for handling infrequent critical events occurring in dynamic environments.
- Train users to recognize inappropriate uses of the aid and to recognize errors. Provide readily accessible lists of limitations; include information concerning limitations and errors in embedded training. Users should learn not to categorically accept a decision aid's capabilities.

### **11.4.5 Decision Graphics and Displays**

- Prepare graphics, textual reports, and input screens in formats familiar to the user. This will facilitate rapid and accurate information-processing. However, the user should be able to control formats or to select from alternate preprogrammed formats.
- Graphics are another important part of the user interface. Graphics help assist the user in visualizing information. However, guard against inaccurate graphics, as they can have a strong negative impact.

- Provide historical displays of comparative cases, to include time-sequenced presentations.
- Use spatial rather than textual formats when the task involves extensive spatial processing, in particular when task performance time is limited. Use tables rather than graphs when reading specific data points.

## **11.5 ORGANIZATIONAL FACTORS**

### **11.5.1 Information Requirements**

Ensure that decision aids are flexible in meeting the different information requirements at different organizational levels.

- Different levels of organizations require different levels of abstraction. Ensure that decision aids accommodate different levels of detail and time constraints at each echelon.
- Ensure that command and control decision aids are distributed (i.e., they should support multiple, cooperating decision-makers at different locations sharing a common database).
- Where practical, design decision aids to support the entire command and control process, rather than to support isolated phases of the process.

### **11.5.2 Entire Organization**

Ensure that decision aid designs consider impacts on the entire organization, particularly where organizational goals may supersede those of subordinate decision aid users.

### **11.5.3 Complementary**

Ensure that decision aids complement existing tasks and information-distribution systems.

## **11.6 FLEXIBILITY**

### **11.6.1 Change-over Time**

Design decision aids as adaptive systems (i.e., they must accommodate growth and evolve over time to meet changing conditions, doctrine, etc.).

- Establish policies for implementing changes, as well as the mechanisms for those changes.
- Adjust to changing situations and user preferences (different circumstances and users may require different methods).

### **11.6.2 Maintainability**

Ensure that decision aids are maintainable by the user. Rules, data, and decision logic should reflect current needs.

### **11.6.3 Type of Support**

Allow the user to tailor the type of support provided by the decision aid in the presence of changing conditions.



## REFERENCES

Paragraph	Reference
11.1.1a	Zachary (1988) pp. 997-1030; McKeown et al. (1991); Ehrhart (1990); Andriole and Adelman (1990)
11.1.1b	Andriole and Adelman (1990); Zachary (1988) pp. 997-1030; Walrath (1989)
11.1.1c	Zachary (1988) pp. 997-1030; Walrath (1989); Andriole and Adelman (1990); McKeown et al. (1991); Ehrhart (1990)
11.1.1d	McKeown et al. (1991)
11.1.2	Zachary (1988) pp. 997-1030; Lysaght et al. (1988); McKeown et al. (1991)
11.1.3.1	Holtzman (1989); Lysaght et al. (1988); McKeown et al. (1991); Walrath (1989)
11.1.3.2	Walrath (1989); McKeown et al. (1991); Holtzman (1989)
11.1.3.3	McKeown et al. (1991); Walrath (1989); Lysaght et al. (1988); Klien and MacGregor (1988)
11.1.3.4	Holtzman (1989); Klien and MacGregor (1988); McKeown et al. (1991); Walrath (1989)
11.1.4.1	Holtzman (1989)
11.1.4.2	Holtzman (1989)
11.1.4.3	McKeown et al. (1991)
11.1.4.4	McKeown et al. (1991)
11.1.5.1	Walrath (1989)
11.1.5.2	Walrath (1989)
11.1.5.3	Walrath (1989)
11.1.5.4	Walrath (1989); Gordon (1988) pp. 55-59
11.2.1a	Gordon (1988) pp. 55-59; Thierauf (1988); Main and Paulson (1988)
11.2.1b	Thierauf (1988)
11.2.1c	Main and Paulson (1988)

## REFERENCES (cont'd)

Paragraph	Reference
11.2.1d	Gordon (1988) pp. 55-59
11.2.2	Thierauf (1988)
11.2.2a	Thierauf (1988); Holtzman (1989); Schwartz (1983) pp. 13-17; Andriole and Adelman (1990); LeMay (1988) pp. 227-229; Finke and Lloyd (1988) pp. 170-193
11.2.2b	Urban (1990); Andriole and Adelman (1990); Thierauf (1988); McCann (1988); Main and Paulson (1988)
11.2.3	Urban (1990); Andriole and Adelman (1990); Thierauf (1988); Walrath (1989); Tannenbaum (1990) pp. 54-59; Bidgoli (1989) pp. 27-34; Zachary (1988) pp. 997-1030; O'Keefe (1989) pp. 217-226; LeMay (1988) pp. 227-229
11.2.4a	Minasi (1990) pp. 13-15; O'Keefe (1989) pp. 217-226
11.2.4b	Holtzman (1989); Schmitz et al. (1990) pp. 29-38; Pew (1988) pp. 931-940; Holtzman (1989)
11.2.4c	Holtzman (1989); Bidgoli (1989) pp. 27-34; Thierauf (1988); Pew (1988) pp. 931-940; Finke and Lloyd (1988) pp. 170-193
11.2.4d	Andriole and Adelman (1990)
11.2.4e	Finke and Lloyd (1988) pp. 170-193
11.2.4f	Finke and Lloyd (1988) pp. 170-193; Minasi (1990) pp. 13-15; Tannenbaum (1990) pp. 54-59
11.3.1a	Lysaght et al. (1988); McKeown et al. (1991); Schmitz et al. (1990) pp. 29-38; Pew (1988) pp. 931-940; Ma et al. (1989) pp. 996-1012
11.3.1b	Holtzman (1989); McKeown et al. (1991); Gordon (1988) pp. 55-59; Riedel (1988); McCann (1988); Zachary (1988) pp. 997-1030; Thierauf (1988); Walrath (1989); Ehrhart (1990); Pew (1988) pp. 931-940
11.3.1c	McKeown et al. (1991); Osborn and Zickefoose (1990) pp. 28-35; Lysaght et al. (1988)
11.3.1d	Holtzman (1989); Riedel (1988)
11.3.2a	Gordon (1988) pp. 55-59; Riedel (1988); McCann (1988); Bidgoli (1989) pp. 27-34

## REFERENCES (cont'd)

Paragraph	Reference
11.3.2b	Gordon (1988) pp. 55-59; McCann (1988); Urban (1990); Zachary (1988) pp. 997-1030; Minasi (1990) pp. 13-15
11.3.3a	Lysaght et al. (1988)
11.3.3b	McCann (1988); Ehrhart (1990); Holtzman (1989)
11.3.4a	Pew (1988) pp. 931-940; Ehrhart (1990); Schwartz (1983) pp. 13-17; Riedel (1988)
11.3.4b	Holtzman (1989); Gordon (1988) pp. 55-59; Pew (1988) pp. 931-940
11.3.4c	Holtzman (1989); Ehrhart (1990); Pew (1988) pp. 931-940; Tannenbaum (1990) pp. 54-59
11.3.4d	Holtzman (1989); Ehrhart (1990); Finke and Lloyd (1988) pp. 170-193; Riedel (1988)
11.3.5a	Riedel (1988); Crolotte et al. (1980) pp. 1216-1220
11.3.5b	Riedel (1988)
11.4.1	McCann (1988)
11.4.1a	McCann (1988)
11.4.1b	McCann (1988)12.4.1.3; McCann (1988)
11.4.2	Weingaertner and Levis (1988) pp.195-201
11.4.3	Weingaertner and Levis (1988) pp. 195-201; Riedel (1988)
11.5.1	Thierauf (1988); Schmitz et al. (1990) pp. 29-38; Gordon (1988) pp. 55-59; McCann (1988)
11.5.1a	McCann (1988)
11.5.1b	Holtzman (1989)
11.5.2	Bidgoli (1989) pp. 27-34
11.5.3	Urban (1990); Gordon (1988) pp. 55-59
11.6	Mittal (1985) pp. 32-36
11.6.1	Pew (1988) pp. 931-940



## REFERENCES (cont'd)

### Paragraph

### Reference

- |         |   |
|---------|---|
| 11.6.1a | Osborn and Zickefoose (1990) pp. 28-35; Mittal (1985) pp. 32-36;<br>Schwartz (1983) pp. 13-17; Riedel (1988); McCann (1988);<br>Bidgoli (1989) pp. 27-34; Zachary (1988) pp. 997-1030 |
| 11.6.1b | Pew (1988) pp. 931-940  |
| 11.6.2  | Zachary (1988) pp. 997-1030   |
| 11.6.3  | Barnett (1990) pp. 1552-1556  |

## 12.0 QUERY

A Database Management System (DBMS) is composed of computer software that facilitates processing information into organized or summarized groups. A database consists of interrelated data that are searchable by a computer. Retrieving information from a database using the DBMS involves identifying a set of items that match or are similar to the user's query or statement of information need. The term "data access" refers to the process of locating and retrieving requested sets of data. By contrast, the term "data presentation" refers to the process of displaying that data to the user in an appropriate fashion. Data access is the query, and data presentation is the result.

The software that makes up the DBMS user interface usually consists of applications programs, report program generators, and query languages. The applications programs allow the end users to enter, retrieve, and update the data in the database. Report-generator utility programs help users specify the content and format of reports. Query languages are used to meet requests for information or to provide a means to browse through the database.

Databases are usually searched in a series of steps. The computer-readable message containing the search terms and logical operators for combining them must be derived from the search query or queries submitted by the user. The search terms are then matched against terms in the database file, either indirectly by searching the index or by directly searching records. The computer responds with counts of retrieved items and should allow the user to sample the items by displaying them on the screen. The user can then make iterative adjustments, either to broaden or narrow the scope of the query.

This section initially reviews types of database queries and methods used to store data in databases. Then, the section provides general guidance on user-oriented database design. The remainder provides specific guidance on query screen designs, user requirements, user-friendliness, database searching, and design requirements for novice and expert user interfaces.

### TYPES OF QUERIES

Users most often communicate with databases by means of command-driven (i.e., query languages), form and menu-driven, natural-language, and icon-based interfaces.

Command languages provide flexibility and relieve the experienced user of the requirement to traverse an entire menu structure to select a command. Users of this type of interface must be familiar with the command language, the steps required for solving problems, and the computer's syntax for accomplishing each step of the process. Structured Query Language (SQL) and Query by Example (QBE) are commonly used languages that perform similar functions.

SQL is a textual language that is becoming a relational database standard. SQL includes table definition, database update, view definition, and privilege-granting, in addition to query

facilities. SQL is often embedded in programs written in other languages, where it generates query results that can be processed by programs written in the host language.

QBE is a table-oriented version of the SQL relational database language and is often supported where SQL is used. QBE provides a pictorial representation of database tables. Symbols placed in the proper table columns specify query selection conditions, grouping, data display, and database updates. Although QBE's tabular format offers advantages to users, it requires user sophistication for effective use.

Query By Forms (QBF) presents the user with data-entry forms that also can be used as templates when developing queries. When accessing data, the user can select one or more of the data-entry fields and enter values, ranges of values, or logical conditions, which are then automatically translated to database queries. Using familiar forms for data entry and search tasks facilitates the user's performance in creating straightforward queries.

Menus provide user-friendly interfaces to command languages, such as SQL. They are designed in a hierarchical or tree structure, which allows the user to proceed step by step through the menu structure to the desired level of detail. Some menu systems allow the user to go directly to a specified level by keyboard command or selecting items from a multi-level menu map. Menu-based query aids offer several advantages. They lead the user through the problem-solving process by indicating which options are available at each point. They are relatively easy for a novice to use, particularly when unfamiliar with the query command structure (low syntactic knowledge) or uncertain how to proceed in solving a particular problem (low semantic knowledge). Menu systems also have disadvantages. Users are forced to make selections from the choices offered by the system and are, therefore, subject to any constraints that might be present. If a user makes an incorrect choice at any level, it can be time-consuming and frustrating to retrace the steps in the menu structure.

Natural language interfaces allow users to formulate queries in their native language (e.g., English, Spanish). These interfaces use a knowledge of syntax (grammar) and semantics (meaning) to interpret queries and translate them into the query language used by the database system. This approach frees the user from learning the usual conventions and rules of query language. Although natural language interfaces offer great potential, they may require considerable user effort in setting up the underlying dictionaries.

Users may directly query icons, maps, schematics and other visual depictions of physical objects by using a pointing device to select the picture or its features in some sequence. Pointing devices (e.g., a mouse, touch-sensitive screen, or trackball) are often used in combination with menus and text-entry screens to formulate queries. Direct interaction with visual representations of physical objects and icons can facilitate human performance.

## **DATABASE DESIGN**

Ease of use and overall performance of a database system depend on its file structure (the manner in which the records are organized in the file or database) and search processes. The

details are chosen by the designer or programmer of the system, often with more concern for the programming aspects of a particular model than for the human performance constraints imposed by that model. The optimum form of information representation will be a function of the task being performed. Unfortunately, current research offers little guidance on how to proceed in database retrieval situations.

Database designs typically use hierarchical, network, relational, or object-oriented models. The hierarchical model represents data in tree structures, and networks represent data as interconnected structures of records linked in one-to-one or one-to-many relationships. Relational databases organize data in tables. Because of its power and ease of use, the relational representation is the prevalent model today and is likely to be the database model of choice in the near future. Object-oriented (sometimes called extended relational) database systems are considered to be part of the next generation of database systems. An object-oriented system represents real-world entities as "objects" that have attributes and defined relationships with other objects.

It is important to recognize that each of these database models can influence the format in which information is presented and the way in which the user can add to, retrieve, or change the information contained in the database. In the end, database models determine the modes of user-database interaction, the format in which the data are presented to the user, and the ease with which a user can acquire information from the database.

## **12.1 GENERAL RECOMMENDATIONS**

### **12.1.1 Ease of Use**

Ensure that a query language or procedure is easy to learn and use. Ease of use and user-friendliness often determine whether the database is used. A program will not be used if it is intimidating, is too difficult, or requires too much effort.

### **12.1.2 Interactive Queries**

Give preference to on-line query over batch or off-line modes because it provides the user the opportunity to interact with the system.

### **12.1.3 User Assistance**

Ensure that an application assists the user in creating complex queries and in narrowing down the search in a step-by-step fashion.

### **12.1.4 Error Detection**

Alert the user to syntax errors in queries and, if possible, to semantic faults (semantic integrity).



### **12.1.5 Minimum Training**

Require only the minimum training. An effective user interface should not require extensive training to be used easily.

### **12.1.6 User-Oriented Designs**

Design the system interface in cooperation with the end users to ensure their satisfaction with the final product. User involvement is most effective when users participate in both developing and implementing the system interface.

### **12.1.7 Multiple Search Options**

Consider the nature of the searches to be performed before choosing an interface format. When more than one type of query is possible, one solution is to choose the interface format that provides the best average performance. Alternatively, provide multiple query and display formats, so the user can change formats as desired or when the nature of the search task changes.

### **12.1.8 Appropriate Displays**

Ensure that displays are appropriate. The forms of information display that facilitate quick responses are not necessarily the same forms that produce accurate responses. The three basic forms of information display are spatial, verbal, and tabular formats. Pictures (spatial) are superior to words (verbal) in recall and recognition tasks and often lead to quicker completion times on procedural tasks. However, words lead to greater accuracy in performance.

### **12.1.9 Individual Preferences**

Ensure, if feasible, that the DBMS is consistent with user expectations. Individual preferences play an important role in the effectiveness of any query application. Users perform better and provide a higher proportion of correct answers when the format of the database matches the format they prefer. Observation shows that, although experience with an application can lead to changes in preference, only preexisting preferences for display formats influenced user performance.

### **12.1.10 Displaying Results**

Display data numerically or graphically. Graphical displays include the bar graph, plot, pie chart and other computer-drawn pictures. Because graphical presentations provide less accuracy than numerical presentations, the most important consideration is the transfer of meaning to the user (see Figure 12-1).

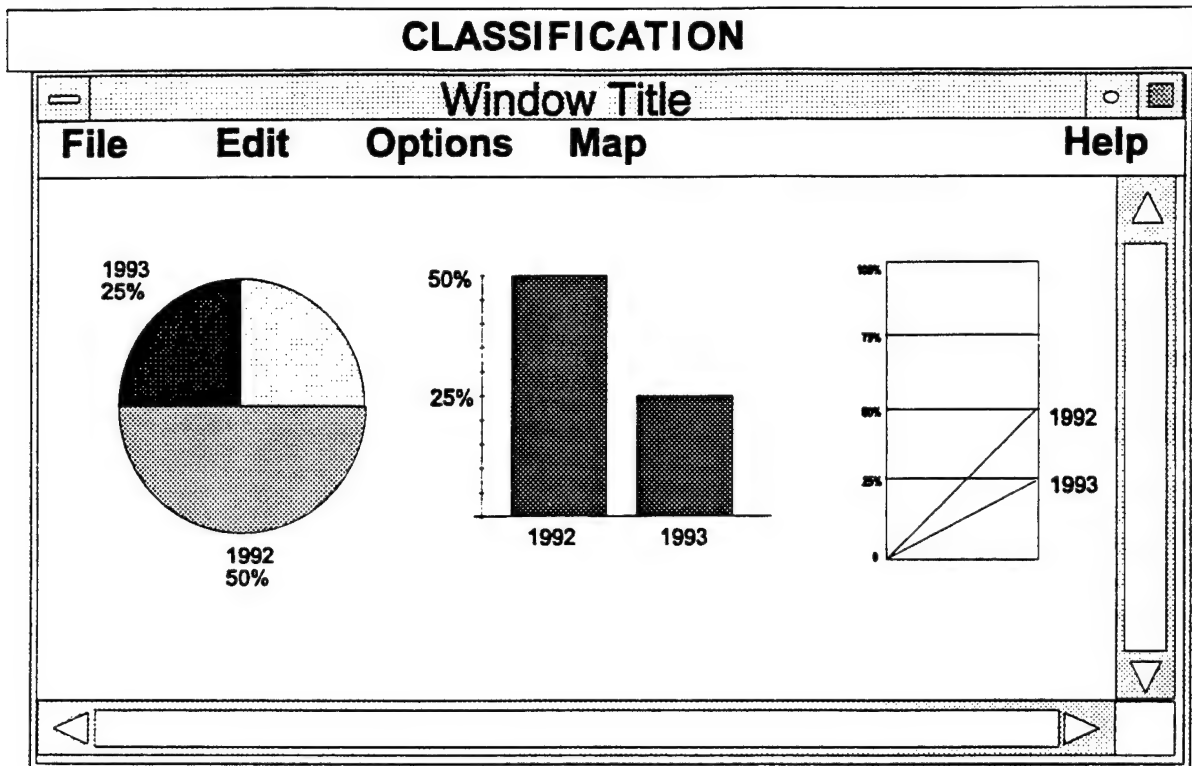


Figure 12-1. Examples of Pie Chart, Bar Graph, and Line Graph

## 12.2 QUERY SCREEN DESIGN

Query screens display the results of a query request or the contents of computer files. The objective in query screen design is to aid the user in quickly and easily locating data or information. Query screen development should optimize human scanning, as scanning is easier when eye movements are minimized, required eye movement direction is obvious, and a consistent pattern is followed.

### 12.2.1 Screen Design Principles

- Include on a query screen only information that is relevant to that screen. Forcing a user to wade through volumes of data is time-consuming, costly, and error-prone. If information will never or very seldom be used, do not display it. An item may be relevant one time a screen is displayed but irrelevant another. Limit a transaction or screen to whatever is necessary to perform actions, make decisions, or answer questions.
- Ensure that the interface display groups information in a logical or orderly manner. Locate the most frequently requested information in the upper left corner.
- Locate the most frequently requested information on the initial screens for multiscreen transactions.

- Ensure that the screen is not overloaded, and use spaces and lines to balance the screen perceptually.
- Use consistent terminology, commands, formats, and general appearance throughout the interface. Ensure that learning can be transferred between modules of the program.

### **12.2.2 Query Screen Organization**

Organize the query screen in a logical, orderly, and meaningful manner. When information is structured consistently with a person's organizational view of a topic, that person comprehends more information. Finding information on a query screen can be accelerated by many factors, including the following:

- The interface should locate the most frequently sought information on a screen in the upper left-hand corner. If there are multiple screen transactions, locate the most frequently sought information on the earliest transaction screens.
- To aid the user in locating a particular item, provide easily scanned and identifiable data fields. Accomplish this through columnization with a top-to-bottom, left-to-right orientation, which permits the eye to move easily left to right across the top of the columns to the proper column before beginning the vertical scan.
- Top-to-bottom scanning will minimize eye movements through the screen and enable human perceptual powers to be used to the fullest.
- Current technology presents query output mainly in tabular format. Emerging object-oriented technology will provide different ways to present such information visually.

### **12.2.3 Captions (Labels)**

- Captions should be complete and written in clearly understandable language. Display captions in upper case, although lower case may be used for long, descriptive captions. Do not use reverse video or highlighting for labels.
- For single fields, locate the caption to the left of the entry fields. Separate the caption from the entry field using a unique symbol and one blank space (a colon ":" is recommended). With multiple occurrence fields, locate the caption one line above and centered over the column of data fields.

### **12.2.4 Data Fields**

- The application interface should ensure that data fields are visually distinct from other displayed information (e.g., field labels).

- The interface should display directly usable information, as well as fully spell out codes and compressed information. The data displayed should include natural splits or predefined breaks.
- The interface should display data strings of five or more characters (numbers or alphanumeric) with no natural breaks, in groups of three or four characters with a blank or other delimiter between each group. Data strings should be left-justified, and numeric data should be right-justified or justified about the decimal point. For all types of data, identical data should be consistent despite their origin (see Figure 12-2).

### 12.2.5 Data Organization

- Organize data in accepted and recognizable order, with vertically aligned captions and data fields in columns.
- Ensure the application justifies data displays consistently.

Poor	Good
Washington DC Cars People Airports	Washington DC Cars People Airports
400 4210 39 39111	400 4210 39 39111
1.5 10.35 1.335	1.5 10.35 1.335

**Figure 12-2. Data Layout and Justification**

- Promote readability by designing the interface with at least one space between the longest caption and the data field column, and with at least one space between each heading. Section headings should be on-line above related screen fields, with captions indented a minimum of five spaces from the beginning of the heading and fully spelled out.

- When presenting multiscreen transactions, place a screen identifier or page number in the upper right-hand corner of the display (i.e., "screen 2 of 5").
- Locate error and status messages consistently in a separate area of the screen. Emphasize these messages by using a contrasting display feature (e.g., reverse video, highlighting, or preceding series of unique symbols, such as asterisks).
- Provide different forms of information display for different search tasks. For example, the interface should provide a selection of display formats as well as a review format where certain fields of the retrieval records can be reviewed without retrieving the entire record.

## **12.3 USER REQUIREMENTS**

### **12.3.1 Search Enhancements**

- Query optimizers are software procedures that automatically enhance the ability of the database application to execute queries. For example, the computer would initiate the search when the first several characters of the search string were entered to reduce the overall perceived time delay of the search. Use query optimizers to increase the effectiveness of the program, but they should be invisible to the user.
- Allow the user to rank search terms by importance. Then use this ranking in a formula for automatically ranking records by relevance in the retrieval set.
- Provide additional search terms in a retrieval set. For example, use a memo field to list the additional search terms related to a particular field.
- Ensure that the application allows redisplay of results of the previous search without requiring reprocessing.

### **12.3.2 Automatic Functions**

- Provide automatic recognition of spelling variants (e.g., color versus colour).
- Provide automatic recognition of acronyms.
- Provide automatic recognition of variations in romanization (e.g., Peking versus Beijing).
- Provide automatic inclusion of the inverted form (e.g., Newborn Infant to Infant Newborn).
- Ensure that the application automatically removes punctuation from search terms when matching them against search-key values.

### **12.3.3 Word Stemming**

- Ensure the application uses a set of rules for reducing words to their root forms by stripping them of their suffixes (e.g., reduce, reduction, reducing).
- If desired by the user, ensure that the application automatically searches the index for all words containing a given root (e.g., the word “form” is the root of formation, inform, and information).
- Provide rules for exceptions based on the language of the discipline or specialty area.
- Allow truncation. The application should automatically search for all words or phrases that begin with the same character stem (e.g., term for terms, termination, and terminated).

### **12.3.4 Erasing**

- Allow immediate deletion of individual characters or deletion of the entire line of input (provided it has not been processed by the computer).
- Permit deliberate interruption of computer messages or displays without disconnection (break or interrupt key).

### **12.3.5 User Satisfaction**

- User satisfaction with the system can be enhanced by including the factors described in the following paragraphs.
- Provide results in a timely manner. One factor of timeliness is the elapsed time from when the command is sent until a response is displayed (response time). Another is the time required for characters or graphics to appear on the screen or hard-copy device (display rate).
- Ensure the appearance, print format, and organization of output are natural to the user. User-generated report formats aid in matching the appearance of the output to what the user expects.
- Minimize the level of effort required by the user, including the limitations or qualifications that the application places on search output.
- Provide maximum capability to the search system while maintaining maximum retrieval effectiveness. For example, do not increase the database size to the point where retrievals take excessive time without also improving search methodology to compensate.
- Ensure that the application assists the user in formulating searches for maximum usefulness of the search results.

## **12.4 USER-FRIENDLINESS**

### **12.4.1 Commands**

- Use mnemonics to avoid the need for remembering syntax (i.e., as sequences or specifications in output instructions).
- Use commands in an easy-to-learn, user-oriented system language.
- Use unambiguous commands. The meaning should be clear to the user.
- Ensure that entering data is not physically awkward for the user, and keystrokes are limited to those absolutely necessary. Provide the capability to define Ctrl key, Alt key, or function key combinations (i.e., Ctrl/Alt/Del to reboot the system) in place of keystroke combinations.
- When a command will delete stored information, provide a complementary command that reverses the action. If deletion is irreversible, provide the user with the opportunity to reconsider the action. The application should check for meaningless commands against a list of authorized commands, after which the application should allow the user to enter a revised command rather than automatically abort the procedure.
- Provide the user with abort or escape facilities for controlling the dialog flow.

### **12.4.2 Computer Messages**

Messages should be clear, simple, and concise. Present the user with the briefest message that can be properly interpreted. Directive messages should be specific and in the context of the current working environment. Messages should warn the user of irreversible action.

### **12.4.3 Error Messages**

Deal with mistakes in a positive, helpful manner. Users will thus gain confidence in the system and feel less intimidated or fearful of damaging it or the data. The error message should appear when the user enters a command that is misspelled, improperly formatted, or cannot be processed because it is inappropriate to the situation. The message should provide instructions for revising the erroneous command.

### **12.4.4 Documentation**

Full system documentation should be available in manual form.

### **12.4.5 Tailor the Interface**

Tailor the interface to suit the needs of users.

- Tailor frequently used queries. In cases where the value of only one or two parameters changes, provide the user with default values for those parameters. For example, a query might request the names of all *Army officers with over ten years service who graduated from an academy in the top 10 percent of their class and who serve in the infantry*. This query contains elements that could be requested several different times, using slightly different conditions each time.
- Macro definition procedures (user-defined commands) are an important feature for expert users who prefer to define their own commands and personalize their environments by encapsulating frequently used query sequences in a new command. Macros greatly simplify user interactions with the application as well as save time. The application should allow the user to store these macros as files or define function key combinations to perform the function.

#### **12.4.6 Accelerators**

Ensure that the interface provides accelerators to save keystrokes. For example, special keys can be dedicated to commonly used functions. The application should permit direct commands as alternatives to menu options.

#### **12.4.7 Backup**

Ensure that the application shields the user from system failure. Provide backup facilities both internally by the software application program and externally by the operating system.

#### **12.4.8 Restore**

Ensure that the application provides a restore utility to facilitate recovery of damaged or destroyed data from backup copies.

#### **12.4.9 Interrupt**

Ensure that the system provides the capability to interrupt work with the application software, then comes back later to resume work at the same point.

### **12.5 SEARCHING**

#### **12.5.1 Commands**

Make the following types of database utility and search commands available to the user.

- Provide a database SELECT command.
- Provide commands to create and erase sets.
- Allow users to combine two or more sets to create new sets.



- Provide the capability for users to specify report formats. The user should be able to name the report, identify the relations from which the report data will be derived, determine the report layout, and define the lines and headings or captions of the report. The user should be able to save the created formulating query and report format for later use.
- Provide the capability for users to restrict the output of retrieval sets.
- Provide the capability for users to save search results easily.
- Provide the user a list of previous search commands upon request. The number of saved commands could be set by the user or could be a prespecified number.

### 12.5.2 Control Functions

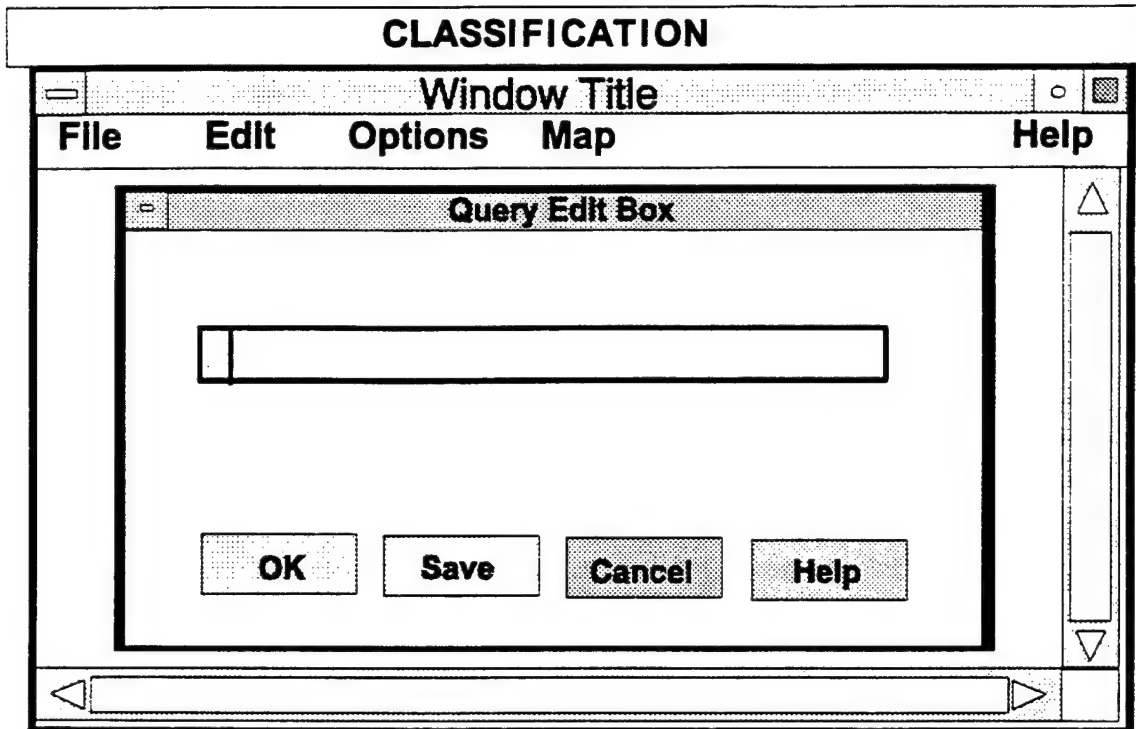
Ensure that the application provides control functions to aid the user in dealing with the system. These functions should include signaling about the system's current state or performing an action based on the state.

- The input parameter for the MARK command should be the current field value, and the application should note the marked value for future reference. For example, fields or records could be marked for deletion.
- DESCRIBE should use as its parameter the current field value. Provide the user with a detailed explanation or description.
- The parameter for the DROP command should be the current field value. The current field should be dropped from the structure.
- The application should provide the user status information upon request. This should include the completion and success or failure of the last search operation executed.

### 12.5.3 Editing Commands

Editing commands are necessary during query formulation. The application should provide a text-editing box to be used for typing search queries. The following functions should be available (see Figure 12-3).

- CUT should allow the user to remove the selected text and place it in a clipboard.
- COPY should allow the user to duplicate selected portions of text and place them in a clipboard.
- PASTE should allow the user to place text from the clipboard into the current text.



**Figure 12-3. Sample Text Editing Box**

- CLEAR should remove all characters currently in the text-editing box.
- SEARCH should allow the user to locate a word or group of characters in the text-editing box.
- When used in conjunction with SEARCH, REPLACE allows the user to replace a word or set of characters with another word or set of characters.
- SPELL CHECK should check the words in the text-editing box against a dictionary of recognized words. This function also should check textual commands to assure correct spelling and syntax.

#### **12.5.4 Query Formulation Commands**

Major tasks performed by queries include extracting, manipulating, and performing calculations on tabular data, including creating tabular results and new tables. Query applications should be able to build functions as needed for developing application programs, as well as update and maintain tables.

- SELECT should provide a means of identifying fields to appear in the query results.
- COMPILE should generate an executable function and check for correctness.

- RUN or DO QUERY commands cause execution of the query. The application should monitor the execution with prompts for input and error recovery.
- The SHOW command should allow various presentations of a tabular result and could be used to present a preview of the results of a query or report.
- MODIFY should allow the user to make changes to the query definition of an already existing query or report. The new query could be saved to a file or as a report, if desired.
- SAVE should allow repeated use or modification of a query. Store the queries in a file with a unique extension, such as ".QRY."

### 12.5.5 User-Friendly Searching

- Abbreviations should be significantly shorter than the original word or mnemonic. Truncation is the preferred form of abbreviation. The application should allow both the abbreviation and the full term.
- The application should automatically complete a search term (opposite of truncation) as soon as it recognizes that the portion of the term entered is unique in the index of search terms. The application should stop the user from typing once the search is uniquely identified.
- Because even simple queries can overload the computer, the computer should inform the user of the problem and prompt user input to terminate the query or continue.

### 12.5.6 Features

- Provide an interactive program that allows the user to navigate through the database. The BROWSE function is especially helpful when queries would be too lengthy to run interactively.
- Provide facilities to format the results of queries as reports.
- Ensure that the application provides the ability to view the list of words and phrases available for searching and term variations, including a link to a database thesaurus to suggest search terms.
- Parsing is the process of deciding how the field will be entered into the search index. Parsing decisions have a direct impact on how a database can be searched. Provide flexibility in searching, regardless of how fields were parsed.
- Use proximity searching, which provides the ability to search words in a positional relationship from word index fields such as titles or abstracts. The words should be either in a specific order or independent of order. For example, the words "query" and "formation" could be searched in the same field.

- Ensure that the application provides the use of Boolean logic, including the use of the logical operators AND, OR, and NOT. It should prompt the user for sets consisting of search terms and combine (intersect) the sets. The search can then be completed as a combined (union) set. The application also should allow interactive editing of queries.
- Ensure that the application provides set building as a means of performing the search in a series of steps, then views the records that answer a query as a set defined by the query.
- Include range searching in the application. This type of search should be based on an ordered sequence using FROM and TO.
- Ensure that the application allows the user to specify the fields to search, because limiting a search to particular fields may speed the search.
- Use a controlled vocabulary of natural language terms. This helps novice users formulate queries.
- Ensure that the application facilitates selecting search terms from key words in records. Then, the interface can display these terms and prompt user selection. For example, the application could rank additional search terms by frequency of appearance in a retrieval set and provide them in ranked order.
- Provide the capability for the system to search on specific data field values input by the user. The application should provide a list of possible field values from which users select.
- Ensure that the application is able to order the field values in a reasonable way, such as alphabetically or from greatest number to the smallest.
- Ensure that the application provides a crossfile search, which will obtain the number of references in all potential databases for the search terms or search profile.

## **12.6 MULTIPLE LEVELS**

User-friendly features and requirements differ for the novice and experienced user. Because the novice will become a more experienced user, the HCI needs to change to suit the evolving needs of the user and the demands of users who have different levels of expertise.

### **12.6.1 Accommodate Novice and Experienced Users**

Multiple levels of interaction are necessary to accommodate the varying levels of experience.

- Users should be able to change levels at any time during a session.
- A tutorial mode should be available when possible.
- Offer context-sensitive HELP on request at all levels.

### **12.6.2 Novice Users**

The level of computer knowledge required of a novice user should be minimal. The application should be easy to use, provide familiar terminology, and allow the user to begin work with little training. To be used effectively, an application should not depend on a complex command language. However, this ease of use may require a loss of power and flexibility.

- An interface for novices may contain only a subset of the search capabilities. This system may be a scaled-down version of a more comprehensive program. In addition, these interfaces may require fewer searchable fields, so the system may not attain the same specificity or variety of search techniques.
- The computer software should prompt the novice user to select from a list of options. The interface should provide an explanation of the options presented.
- The interface for novices should have a simplified command structure using fewer and more easily understood commands.
- Mnemonic selections are preferred over numbered selections.
- The system design should strive for intelligent interfaces between naive users and search systems. Two main components of an intelligent front-end are forms and graphics. Menus and data forms can control the flow of the application, and graphics can be used to provide a visual readout of the data.

### **12.6.3 Experienced User**

Experienced users can accommodate comprehensive versions of query applications.

- The application designed for the expert user can reduce computer overhead by providing less detailed on-line information.
- The application should allow the experienced user to enter multiple commands to speed the dialog.

## REFERENCES

Paragraph	Reference
12.0	Hansen and Hansen (1992); Flynn (1987) p. 221; Harter (1986) pp. 124-127; Frost (1984) p. 8; Humphrey and Melloni (1986) pp. 41-50; Kelley (1984); Katzeff (1986); Martin (1983) pp. 427-483; Hershman, Kelly, and Miller (1979); Schur (1988); Shneiderman in Vassiliou (1984); Boehm-Davis (1989); Ogden and Brooks (1983)
12.1	Martin (1983) pp. 427-483
12.1.1	Tenopir and Lundeen (1988) pp. 43-45
12.1.2	Frost (1984) pp. 187; Martin (1983) pp. 143, 452
12.1.3	Martin (1983) pp. 452
12.1.4	Grill (1990) pp. 78
12.1.5	Cuff (1980)
12.1.7	Boehm-Davis (1989)
12.1.8	Boehm-Davis (1989)
12.1.9	Boehm-Davis (1989)
12.1.10	Flynn (1987) pp. 401-441
12.2	Galitz (1989) pp. 155-165; Flynn (1987) pp. 401-441
12.2.1e	Tenopir and Lundeen (1988) p. 45
12.2.5f	Boehm-Davis (1989)
12.3	Humphrey and Melloni (1986) pp. 200-203
12.3.1a	Chapnick (1989)
12.3.3d	Humphrey and Melloni (1986) pp. 143-144, 200; Tenopir and Lundeen (1988) pp. 34; Ehrenreich (1982)
12.3.5	Harter (1986) pp. 154
12.3.5a	Shneiderman in Vassiliou (1984) p. 5
12.4	Tenopir and Lundeen (1988) pp. 44-45; Humphrey and Melloni (1986) pp. 197-199
12.4.1d	Flynn (1987) pp. 520-521
12.4.1e	Flynn (1987) pp. 514-515

## REFERENCES (cont'd)

### Paragraph

### Reference

- |         |  |
|---------|--|
| 12.4.1f | Benbasat and Wand (1984); Carlson and Metz (1980)  |
| 12.4.4  | Tenopir and Lundeen (1988) pp. 155-167   |
| 12.4.5  | Frost (1984) pp. 241-242; Feldman and Rogers (1982)  |
| 12.5    | Harter (1986) pp. 76-94  |
| 12.5.1  | Harter (1986) p. 28  |
| 12.5.2  | Lochovsky and Tsichritzis in Vassiliou (1984) p. 131   |
| 12.5.4  | Schauer in Blaser and Zoeppritz (1983) p. 33   |
| 12.5.5  | Humphrey and Melloni (1986) p. 200; Tenopir and Lundeen (1988) p. 34; Ehrenreich (1982)  |
| 12.5.5c | Grill (1990) p. 78   |
| 12.5.6  | Tenopir and Lundeen (1988) pp. 31-39   |
| 12.5.6a | Frost (1984) pp. 196-197   |
| 12.5.6c | Sormunen in Wormell (1987)   |
| 12.5.6j | Humphrey and Melloni (1986) p. 200; Harter (1986) pp. 41-58; Ogden and Brooks (1983)   |
| 12.5.6k | Humphrey and Melloni (1986) p. 202   |
| 12.5.6n | Sormunen in Wormell (1987)   |
| 12.6    | Humphrey and Melloni (1986) pp. 199-200; Benbasat and Wand (1984); Tenopir and Lundeen (1988) p. 45; Shneiderman in Vassiliou (1984) pp. 4, 7; Harter (1986); Schur (1988); Kelley (1984); Flynn (1987) pp. 504-506; Cuff (1980); Sormunen in Wormell (1987) |

## 13.0 EMBEDDED TRAINING

The interface of the optimally designed computer program should be designed and tested such that no user assistance is needed. However, because of differences between humans and computers, the variety of task demands, and the ever-present human tendency to make errors, assistance is needed. People differ in computer experience, patience level, learning style, reasoning ability and style, and numerous other characteristics. At the same time, sophisticated computer systems and software programs are often highly complex but still retain the requirement to be highly usable without requiring extensive training or technical expertise. Assistance programs offer one of the primary methods used by designers to achieve a high degree of usability.

User assistance is commonly offered through on-line help, documentation, and on-line training. The distinction between on-line help and on-line training is often blurred. For the purposes of this *Style Guide*, on-line help refers to assistance for a specific problem, function, command, or term. On-line training programs focus on process; they offer instruction.

On-line training programs may exist completely embedded within the application software, separately as an application, or as a combination of both. The on-line training program also may be executed by some form of supplemental component (e.g., strap-on [video disk player] or plug-in [floppy disk]). Though many guidelines apply to both embedded and supplemental training, interface guidelines presented in this section pertain specifically to embedded training.

The guidelines also apply to a range of embedded training formats and capabilities including:

- Fixed format provides the same information regardless of what the user has done.
- Context-sensitive format depends on what users are currently trying to do or on the context in which they are working
- Prompting intervenes or prompts automatically if a user proceeds incorrectly.
- Dialog allows users to obtain assistance through natural language interaction.
- Adaptability keeps track of a user's operation and provides appropriate help or training based on the user's operation, for example, intelligent tutoring systems (Kearsley 1986).

The embedded training guidelines included in this section are derived from the results of empirical research, reported computer training experience, and experts' recommendations. Guidance for embedded training interface design appears under a variety of types of on-line training: Computer-Assisted Instruction (CAI), Intelligent Computer-Assisted Instruction (ICAI), Computer-Based Training (CBT), Intelligent Tutoring Systems (ITS), Embedded Training (ET), coaching, Electronic Performance Support Systems (EPSS), and guided discovery, among others.



On-line training strives to support learning how to use an application. However, it conflicts with the user's primary task, because consulting a training program interrupts work in progress. This conflict may cause new users to skip training altogether or to select immediate task help without furthering their overall understanding of the system (Grice 1989; Hackos 1991; Horton 1990). Two crucial factors in determining whether or not users accept and use embedded training are a well designed, intuitive interface and the opportunity to practice. Each of the embedded training guidelines addressed in this section assumes a basic set of objectives for assisting users: consistency, efficient use of capabilities, minimal memory load on users, minimal learning time, and flexible support of different users.

The goal of embedded training interface design is to ensure users can obtain answers to their questions with maximum efficiency, maximum accuracy, and minimum additional memory requirements. An embedded training program should answer the following types of questions:

- Goal-oriented: What types of things can I do with this program?
- Descriptive: What is this? What does this do?
- Procedural: How do I do this?
- Interpretive: Why did that happen? What does this mean?
- Navigational: Where am I?
- Choice: What can I do now?
- History: What have I done? (Baecker and Small 1990; Gery 1991; Laurel 1990).

The manner in which assistance is provided affects the ability of users to learn and transfer that learning to other situations. Research in instruction and on-line documentation has identified basic concepts and practices that support learning and transfer. Central among these concepts relating directly to embedded training are:

- Opportunity to practice
- Readability
- User control - perceived and actual
- Learning mode - visual (graphics and text)
- Advance organizers.

Specific guidelines related to these concepts appear in embedded training components and instructional presentation guidelines (see Subsections 13.4 and 13.6).

Much of the relevant research and development work from which these guidelines were developed comes from individual demonstration and limited distribution systems. Significantly less empirical research has explored the behavioral issues pertaining to on-line training or advice-giving systems. On-line training experts suggest additional behavioral research, including effects of feedback timing, preferences for and effectiveness of different on-line training components, suitability of media presentation (animation, text, sound), and the effects of system-initiated intervention. The current research focus has moved from building systems capable of detecting all possible errors and misconceptions to building an empathetic partner that chooses among several forms of interaction based on the content of the task and needs of the user.

## **13.1 GENERAL**

A strong embedded training interface provides users with an understanding of the training program and the linkage between the application program and the training program. The guidelines in this section address user orientation and the linkage between application and embedded training programs. Section 13.0 of the *Style Guide* includes general guidance for embedded training, followed by guidelines pertaining to more specific features. This section also contains a series of figures illustrating embedded training. Each figure is based on a basic screen prototype (see Figure 13-1). To illustrate a particular guideline clearly, a number of the figures show only a portion of the basic screen display.

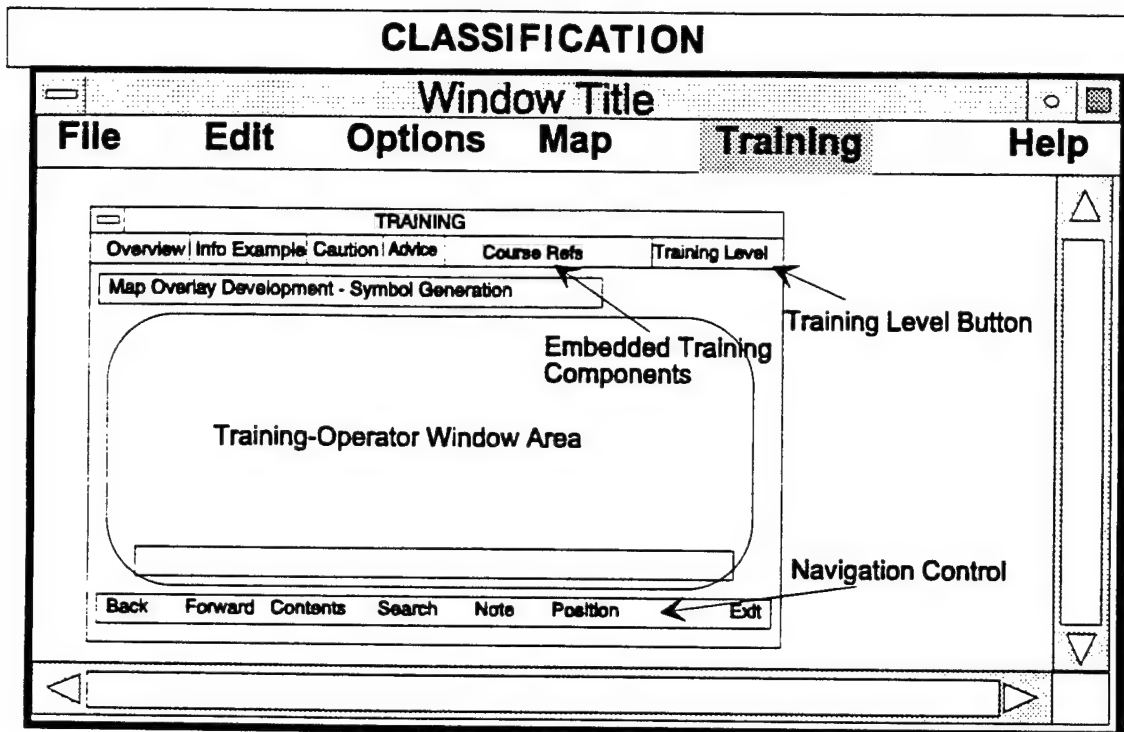
### **13.1.1 Initial Use Overview**

Provide first-time users of embedded training an overview of the embedded training program. This orientation should convey what the embedded training achieves by combining text and graphics (animated or static).

### **13.1.2 Positive User Attitude**

Build positive user attitudes and increase use of the embedded training by ensuring that the interface maintains a positive tone and does not evaluate the user's performance when practicing and experimenting. Ensure the system messages do not blame the user and avoid implying that the computer is human. For example, "You can use the training Program to learn..." is preferable to, "The training Program can teach you..."

- Do not use personalized messages, as they interrupt and often annoy users.
- The effectiveness of the embedded training is related directly to the accuracy of the embedded training information.
- Avoid personalization (i.e., "You did a good job, Sam") and personal recognition (including even simple statements, such as "Excellent!").



**Figure 13-1. Simplified Prototype Embedded Training Screen**

### **13.1.3 Availability of Embedded Training**

Provide embedded training programs to users at all points during use of the application, except where it would interfere with time-critical operations.

### **13.1.4 Accuracy of Embedded Training**

Ensure embedded training is accurate, reflects the most current form of the application, and is updated in response to changes in the application. When changes occur in application procedures or critical operations, ensure that users are notified. In addition, consider providing users with an option to see new and revised information (i.e., by selecting "News").

### **13.1.5 Moment of User Need**

Provide training support at the moment the user needs it whenever possible.

### **13.1.6 Embedded Training Browsing**

Allow users to work with the embedded training independent of the application, to accommodate user browsing.

### **13.1.7 Return to the Application From Embedded Training**

Ensure users can return to the application from any point within the embedded training with a single action (e.g., keystroke, command, point and click) without shutting down either system.

### **13.1.8 Application Restore Screen**

When users exit the training program, restore the application screen to the state that existed prior to the request.

### **13.1.9 Restore Embedded Training**

When training is interrupted (e.g., system failure, time-critical requirements) or users exit before completion, offer them the opportunity to return to the position in the embedded training that existed before the interruption.

### **13.1.10 Protection From Hazardous or Destructive Actions**

Prohibit users from accidentally activating hazardous events (e.g., mine field activation) and destructive control actions (e.g., accidental erasure or memory dump) during the embedded training.

### **13.1.11 Application Screen Protection**

Ensure embedded training commands do not alter or destroy application screen data.

### **13.1.12 Noninterference During Critical Operation**

Prohibit system-initiated embedded training interruption of the primary application during a critical operation.

### **13.1.13 Notification of Critical Operation**

Ensure the user is notified of incoming critical application information (e.g., tactical operation input).

### **13.1.14 Multiple Stations**

If the application system has multiple stations, ensure that stations using the embedded training have no effect on the stations performing an operational task.

### **13.1.15 Context Sensitivity**

Make the training context-sensitive; that is, wherever possible, the training should depend on where the user is in the application or on the general nature of the content of the application.

### 13.1.16 Consistent Application Interface

Provide the greatest possible consistency between the application interface and the embedded training interface to ensure a smooth transition between platforms and to minimize the user's learning requirements (e.g., terminology, displays, commands).

### 13.1.17 Inconsistent Interface Assistance

Provide assistance if the embedded training interface is substantially different from the application systems operations or when the embedded training interface has complex features that might need to be explained.

## 13.2 ADAPTATION TO USERS

Users vary in many ways, including computer experience, domain experience (program content - e.g., command and control), learning style, preferred work style, and immediate task demands. Adapting the embedded training interface to the user's characteristics and preferences will encourage use of embedded training and, consequently, should increase user efficiency with the application (see Figure 13-2).

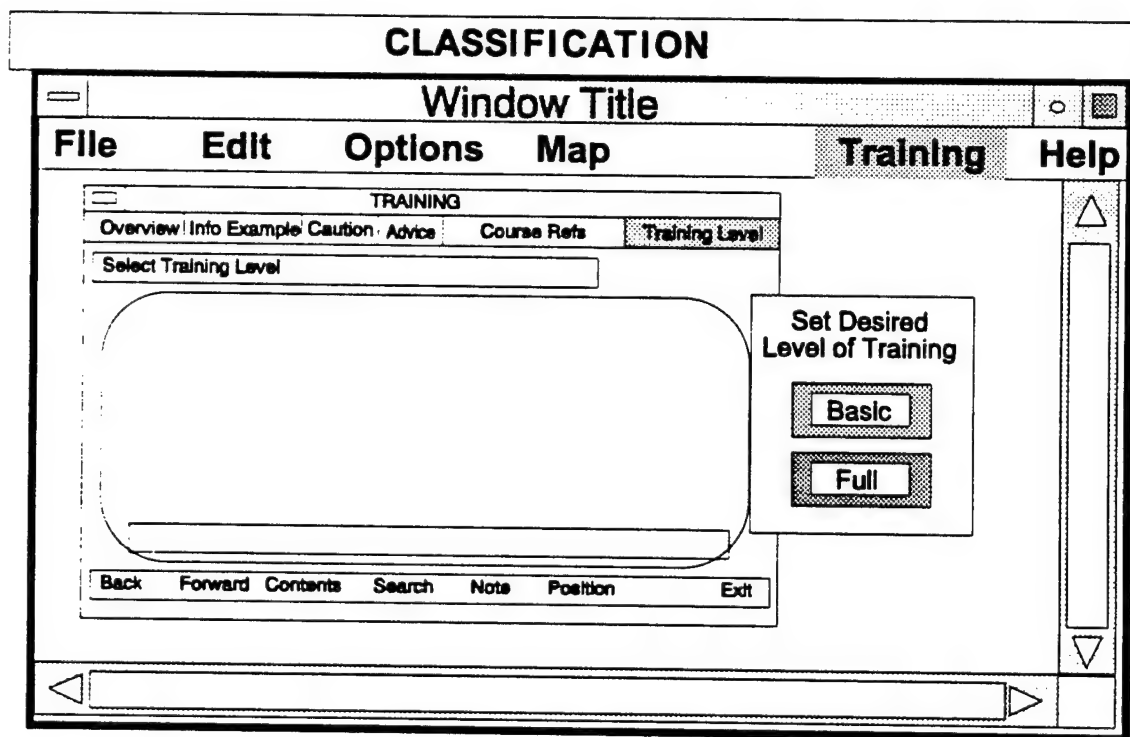


Figure 13-2. User Selection of Training Level

### **13.2.1 User Control Over Level of Difficulty**

Accommodate the differences in computer experience by allowing users to select the level and type of assistance. The ability to select is important because users may be novices in some areas, casual operators in others, and experts in still others. Reducing the complexity of the training interface for beginners simplifies the demands of learning.

- Allow novice users and/or first-time users to select a restricted capability interface that blocks features and allows only basic feature operation (e.g., in word processing - creating, editing, and printing).
- Provide novices only the necessary information, but allow them access to all of the capabilities by direct request.
- Offer experts assistance in using the system more efficiently (e.g., shortcuts, limitations, complex operations).

### **13.2.2 Learning Structure**

Allow the user to select a type of learning structure. This accommodates individual needs for information and practice. Learning structure types may be:

- Discovery - undirected exploration or browsing
- Guided/supported discovery - directed exploration
- Structured - menu identifies options explicitly and provides implicit cues.

## **13.3 EMBEDDED TRAINING COMPONENTS**

An embedded training system can integrate several resource components to support users while they perform their jobs. Embedded training programs may provide an information database (infobase), common errors, examples and scenarios, interactive advice, internal cross-referencing, expert system-initiated training, and formal courseware.

### **13.3.1 Multiple Components**

In addition to the immediate context-sensitive assistance, offer users multi-component training that is easy to specify and access (e.g., scenarios, examples, information databases, off-line references, common problems, and/or coaching).

### **13.3.2 Information Database Component**

Provide users an interactive information database containing both conceptual and task-oriented information.

### **13.3.3 Reference Component**

Provide a reference component that includes all on-line resources, as well as system- and job-related, off-line resources.

### **13.3.4 Examples Component**

Offer users the opportunity to practice using common examples in an exploratory or guided mode, which would allow users to work through the steps required to perform a specific task.

- Encourage user experimentation (i.e., "What would happen if...") by making it easy for them to exit an application, practice, then return to the unaltered application position.
- If users explore a problem within the application, protect the application system with an UNDO command requirement.
- Clearly distinguish between the exercise and the application to minimize possible confusion arising from switching back and forth between operation modes (e.g., highlight or shadow the practice session).
- Avoid demonstrations and exercise summaries if they do not provide opportunities to practice the procedure or function.

### **13.3.5 Advisor or Coaching Component**

Provide an embedded training component that advises or coaches users in solving problems. This may make users aware of enhanced system operation and may also be used in response to user request, system recognition of suboptimal user performance, or complex tasks. A system can coach users through tasks by presenting a series of questions and recommending a course of action based on the responses (see Figures 13-3 and 13-4).

### **13.3.6 Common Errors Component**

Provide users with a context-similar, embedded training component that shows common user errors or "Cautions" associated with a given approach or task procedure (see Figure 13-5).

### **13.3.7 Record Keeping**

Records of user interaction with the embedded training can be helpful to users, supervisors, and system designers. Using embedded training records requires careful planning to avoid threatening users. Users are more likely to experiment and practice if they feel their errors will not be seen by others.

- Allow the user to record the path through a process and/or the training modules completed successfully or unsuccessfully. This will aid in later reference or experimentation.

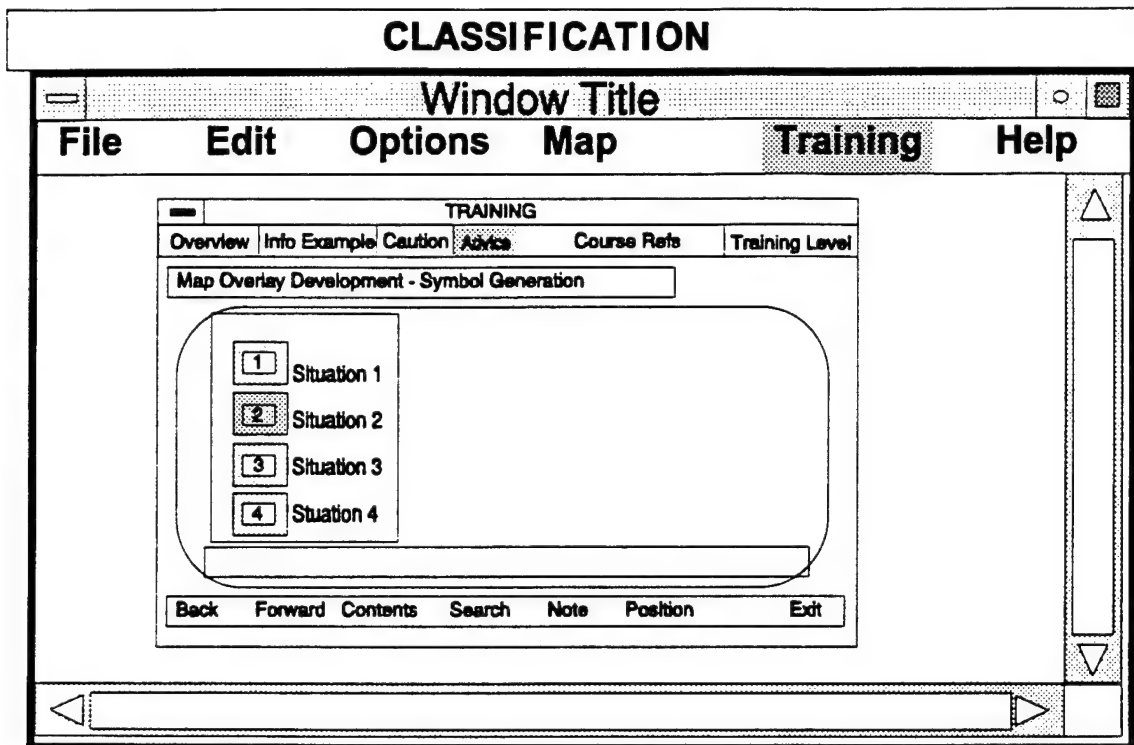


Figure 13-3. Example of How an Advisor Can Guide Users

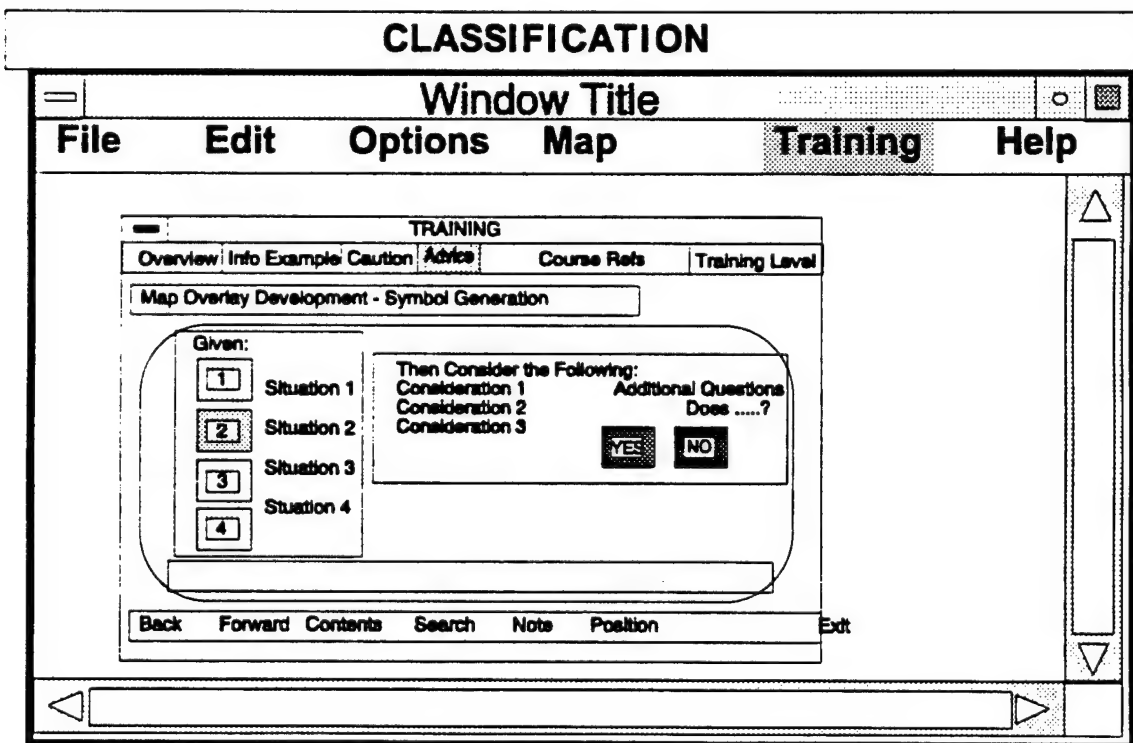
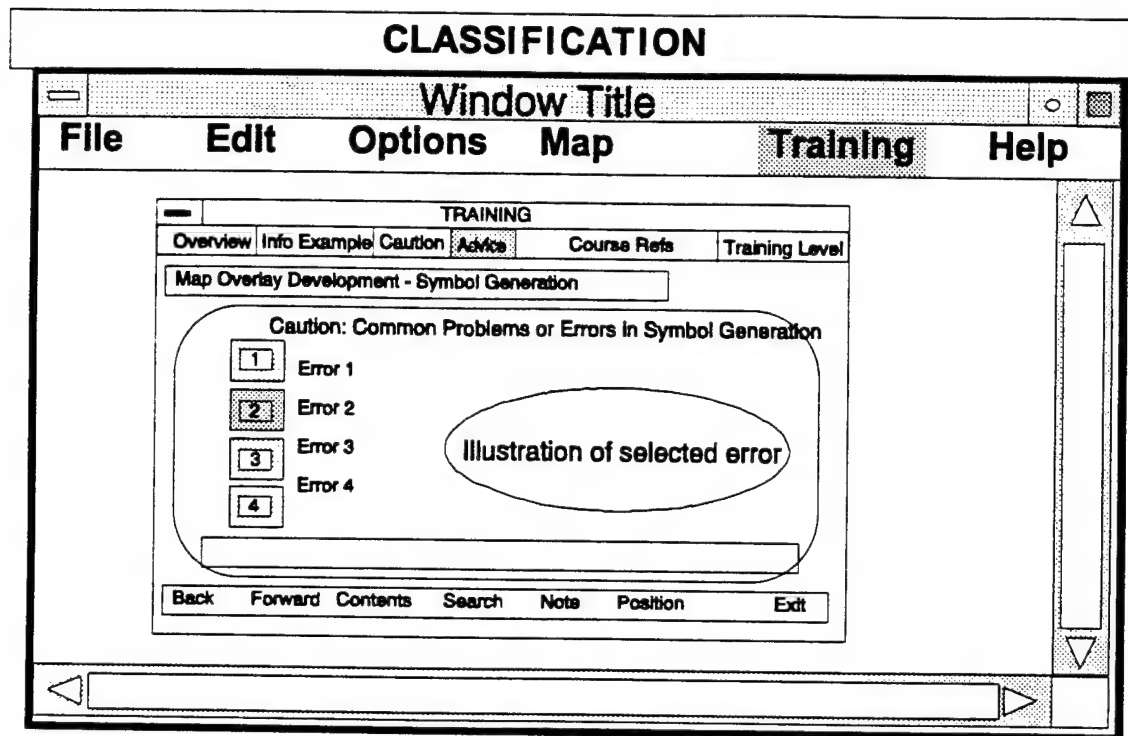


Figure 13-4. Example of How an Advisor Can Guide Users (cont'd)





**Figure 13-5. Example of “Cautions” that Identify Common Errors**

- If records of user training sessions are stored, ensure the privacy of users is protected by storing their records as anonymous files.
- If the exercise or courseware module will be used for evaluation purposes, give users prior notification. Explicitly state the criteria for evaluation.

## **13.4 INSTRUCTIONAL STRUCTURE**

Both the size of instructional unit (granularity) and control of instructional sequence affect the efficiency and attitude of the user.

### **13.4.1 Granularity**

Structure the embedded training components into “single learning episodes,” small enough and homogeneous enough to be learned as single units. This enables users to select the particular section or subtopic within a component for which they desire assistance.

### **13.4.2 System-Controlled Sequences**

For novice users and for embedded training that deals with critical or hazardous procedures, the system should direct user movement through the procedures.

### 13.4.3 Sequence Control for Experienced Users

Provide experienced users with the flexibility to move through the steps of a procedure sequentially or to move directly to any specific step or resource point.

## 13.5 INSTRUCTIONAL PRESENTATION

The manner in which assistance is provided affects the ability of users to learn from the instructional experience and to transfer that learning to other situations. The following statements outline interface guidelines for instructional presentation.

### 13.5.1 Combined Media Presentation

Present the embedded training using a combination of media, graphics, and natural language, where appropriate. Graphic media aid in visualizing significant patterns, whereas natural language text conveys the meaning and significance of the visualization (see Figure 13-6).

### 13.5.2 Graphics for Method-Based Knowledge

Offer users flowchart diagrams that provide an overview conveying method-based knowledge, consisting of a series of procedural steps and decisions.

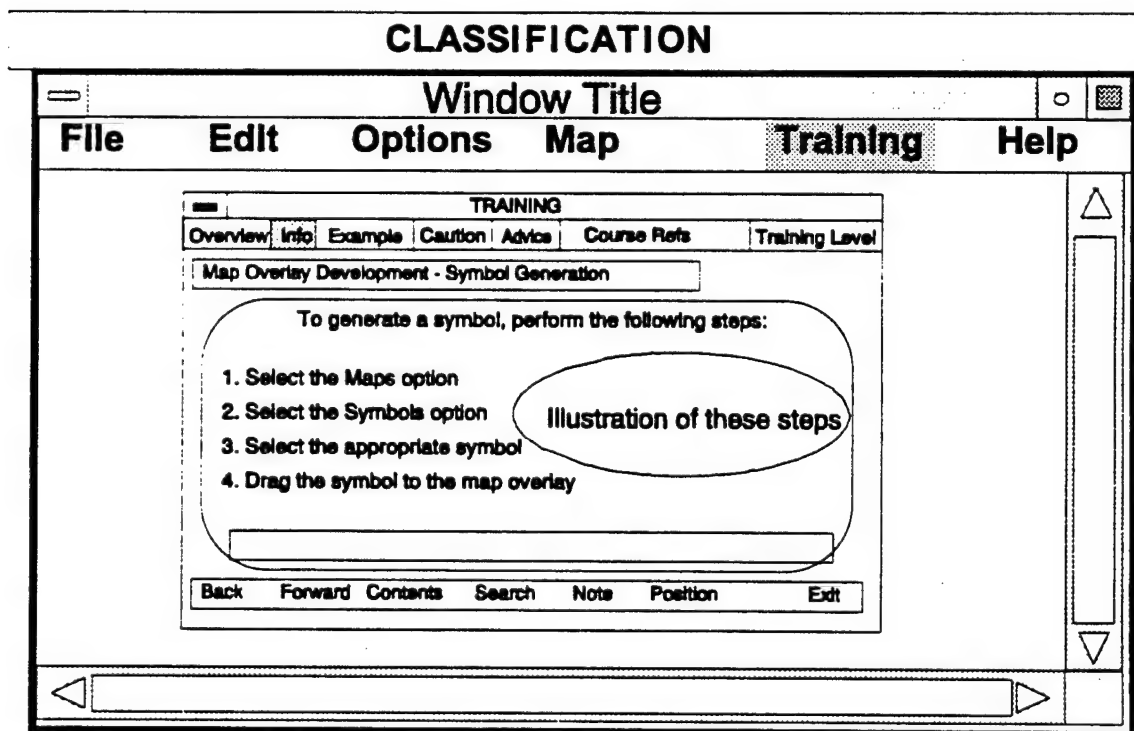


Figure 13-6. Combined Graphic and Natural Language Presentation

### 13.5.3 Reading Requirements

Keep reading requirements to a minimum. Users prefer to read text in print and may not read text on a screen that exceeds even a few sentences in length.

### 13.5.4 Advance Organization

If the component will be used for knowledge training, provide cues, and overviews that orient users unfamiliar with embedded training content and/or process through brief descriptions of scenarios and exercises or courseware outlines. Stated objectives are an important feature for novice users.

- Provide users a brief statement of the exercise objective. The statement should refer to the primary purpose of the embedded training request.
- Clearly identify each embedded training module. State objective, content, and, where appropriate, the number of subsections and estimated completion time.
- Remind user of the purpose of the request for assistance (see Figure 13-7).

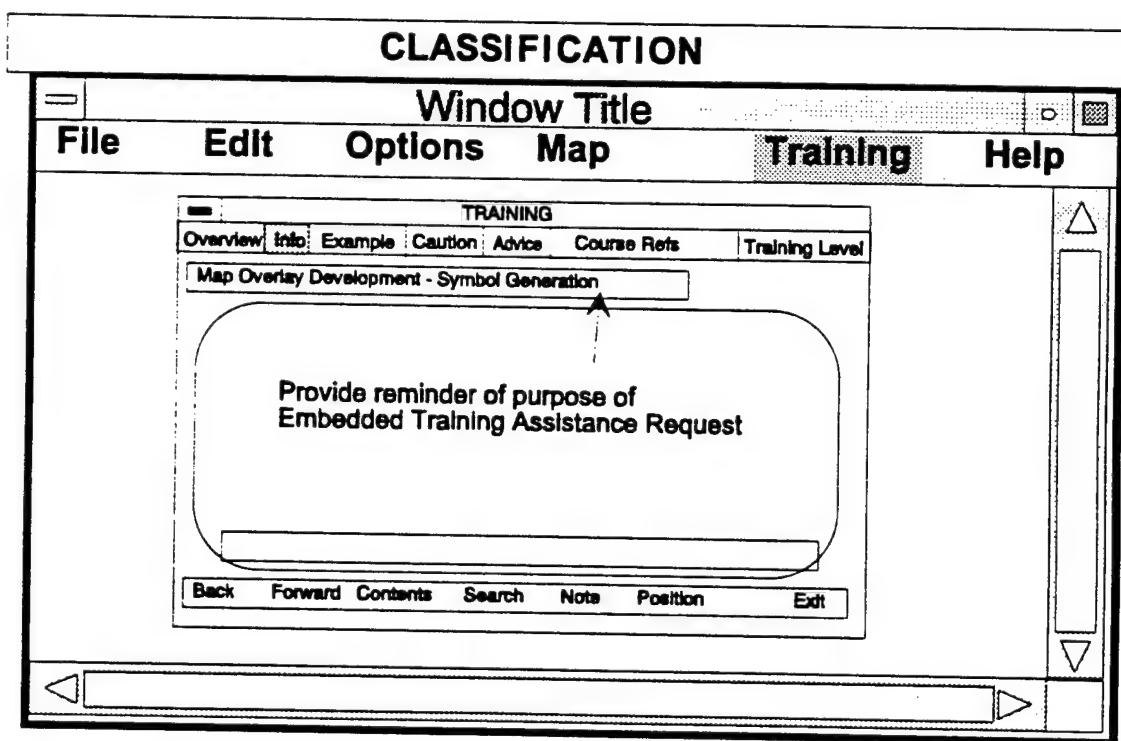


Figure 13-7. Example of Assistance Request Reminder

### **13.5.5 Printing**

Provide users the ability to print embedded training content - ranging from screen displays to courseware, information to study further, or for future reference, and/or to print the displayed training material.

### **13.5.6 Fidelity**

Adjust the level of training content and presentation fidelity to match the training:

- Low fidelity for initial training, simple data, and easy process
- High fidelity for unusual processes, hazardous events, or difficult processes.

### **13.5.7 Simplicity**

Give users simple answers to simple questions. If the answer is long or complex, offer a summary and options to request additional guidance.

### **13.5.8 Verification**

Allow users to verify or confirm selected options, solutions, and commands. This allows them to evaluate the completeness of a process or task and the accuracy of their approach without having to sort through extraneous material.

## **13.6 ACCESSING TRAINING**

### **13.6.1 Displayed Embedded Training Availability**

Display the command, icon, or function key used to access training throughout the application to remind the user of training availability.

### **13.6.2 Access Via Training Icons**

Allow users to access the embedded training directly by selecting an embedded training icon and moving to the point of user need. For example, a user could activate embedded training and select a "Cautions" component icon, move to the point where assistance is needed, click, and receive additional information without exiting the application (see Figure 13-8).

### **13.6.3 Structured Menu**

When using a structured menu to access the embedded training, allow users to add to or change existing embedded training messages (e.g., add terms to the menu using an ADD function). If users are allowed to customize menus, the original menu must be protected (e.g., log-on files for individual users).

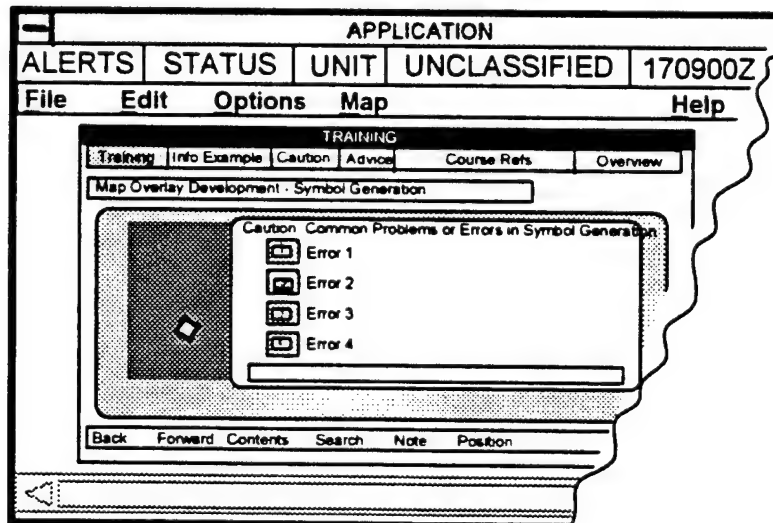
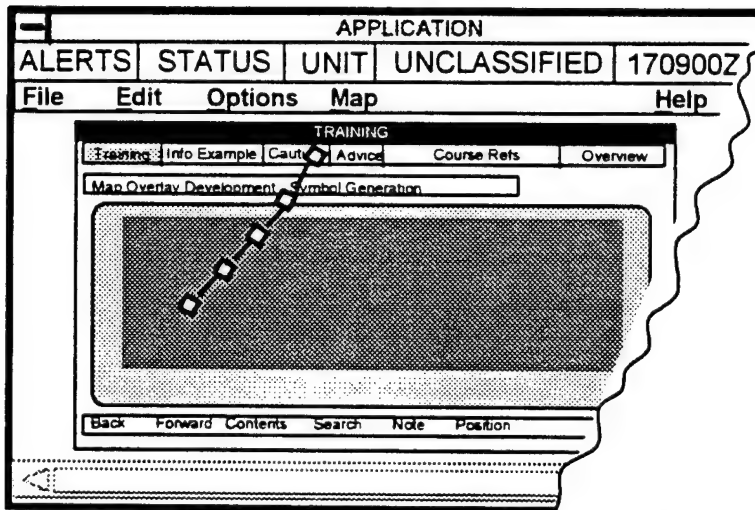
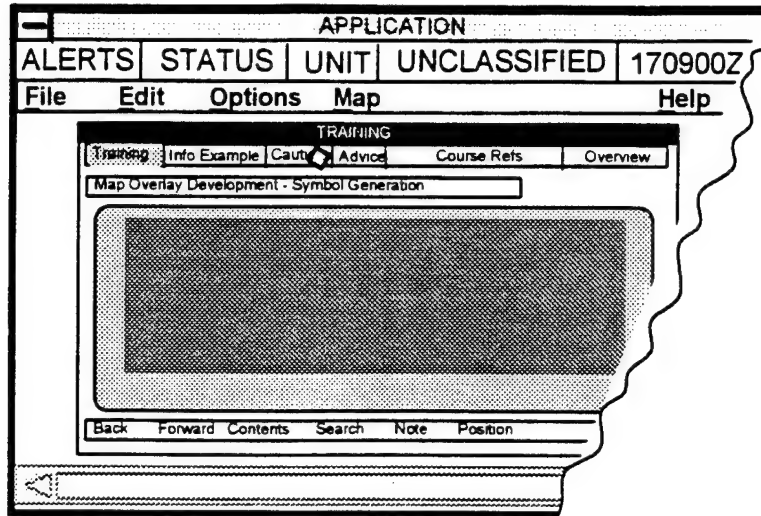


Figure 13-8. Using an Icon to Access Embedded Training Directly

## **13.7 SCREEN DISPLAY**

### **13.7.1 Complete Display**

The content of each screen should stand on its own; do not require users to refer to a previous screen within a module to recall essential information. For example, if users need to enter identical information on a series of screens, the system should automatically enter the appropriate information, repeat the information on each screen, or prompt users to record the information.

### **13.7.2 Graphics**

Select uncomplicated graphics that portray the functional objective clearly, omitting nonessential visual detail.

### **13.7.3 Window Placement**

Display the assistance in windows that do not completely obscure the application's critical navigation buttons, operational icons, or the status message line or window.

### **13.7.4 User Window Control**

Allow users to resize and reposition overlapping windows. This will allow users to see the portions of the application with which they are most concerned.

## **13.8 TECHNICAL COMMUNICATION/WRITING STYLE**

Phrase embedded training topics, messages, and menu options in the active voice. Phrase task-related terms to refer to the learning task (e.g., "Creating and modifying fields" instead of "Fields").

## **13.9 MOBILITY/NAVIGATION**

### **13.9.1 Mobility Within the Embedded Training**

Allow users to move among embedded training components freely:

- Without returning to the top of a central hierarchy
- Without exiting the current embedded training component
- Without having to proceed through a preset path
- Without having to step through introductory material.

### 13.9.2 Embedded Training Navigation Button Display

Display embedded training navigation buttons in each embedded training screen for controlling movement between and within modules (see Figure 13-9).

## 13.10 ERROR FEEDBACK

### 13.10.1 Immediate Feedback

Provide feedback to users in a timely manner, adjusted to the users' expertise.

- When practice requires multiple steps, provide users immediate feedback to avoid a series of incorrect actions.
- For novices and for uncomplicated problems, offer immediate feedback that includes a suggested next best step.

### 13.10.2 Context-Similarity Feedback

Provide feedback to users in a form similar to the application, product, or outcome (e.g., an error in equipment setup may be illustrated by correctly configured equipment rather than by a checklist, menu, or even natural language message).

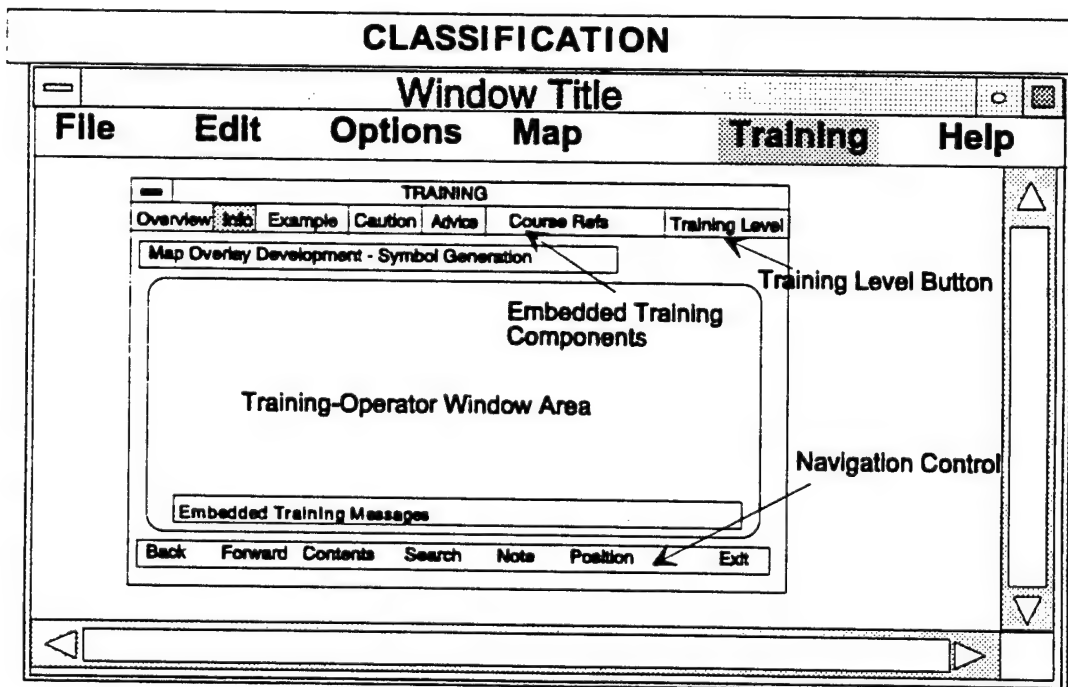


Figure 13-9. Example of Embedded Training Navigation Buttons

### **13.10.3 Error Identification**

Provide specific feedback that identifies errors rather than assigns a score.

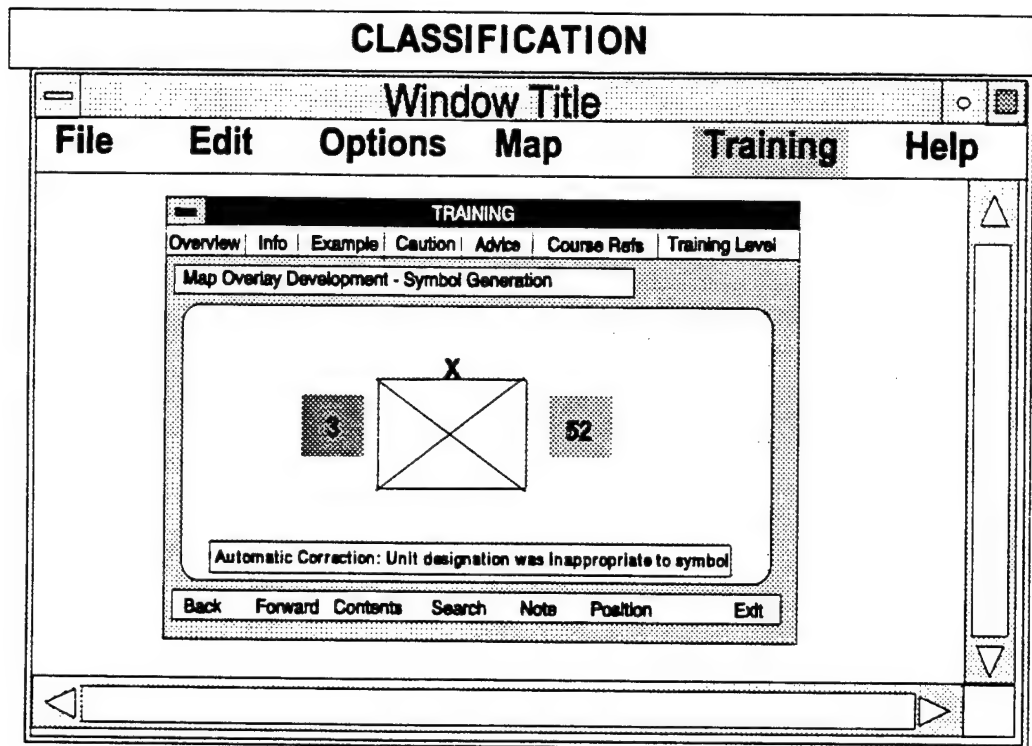
### **13.10.4 Tone of Error Message**

Provide error messages that are constructive and neutral in tone; avoid messages that suggest a judgment of the user's behavior. For example, "the system cannot process..." is preferable to, "Invalid Number: Entry must be 4 digits..."

### **13.10.5 System-Initiated Error Feedback**

- When a response to a system-initiated query or recommended action is required, the system should provide users the opportunity to stop and think (i.e., consider other options, recall past experiences, weigh problem solutions) without premature system interruption. For example, the system should avoid over-prompting and unwanted problem resolution. The system could offer users the ability to place the system initiation on hold or cancel an upcoming intervention.
- Provide novice users with prompts identifying probable next-step errors.
- Use control blocking sparingly (e.g., to protect a system from accidental hazardous actuation and system destruction).
- Allow novice users to select an error-blocking option that limits errors.
- Give users the option to block temporarily the system-initiated error feedback or instruction.
- If the system automatically corrects some errors (e.g., replacing an out-of-bounds parameter), ensure users are notified of the corrections (e.g., by a message and highlighting of the corrected information) (see Figure 13-10).
- Allow experienced users to select automatic system error correction without requiring their confirmation. This option assumes that mundane errors made by experts are a result of minor actions, such as mis-stiking a key, command, or icon.
- Avoid blocking access to system functionality. This can be very frustrating and can be a result of a misdiagnosed error or correct, but uncommon, approach.
- Avoid user confusion that may result from automatic system error correction.





**Figure 13-10. Example of Automatic Correction Notification and Identification**

## **13.11 ABILITY TO MODIFY**

### **13.11.1 Additions to the Embedded Training**

Provide the capability to add to, but not modify, the original training system. If embedded training allows individual user modifications, protect original application and embedded training (e.g., separate log-on files for each user).

### **13.11.2 Multi-User Systems**

On multi-user systems, permit the individual user to store and reference additions in a individual file.

### **13.11.3 Annotation**

Permit users to annotate a copy of the training program (i.e., examples, pitfalls, process notes, references, etc.).

### **13.11.4 Annotation Search**

Provide users the ability to search an annotation log.

### 13.11.5 Icons Used to Designate Annotations

Use icons to designate the position of an annotation in the embedded training program (e.g., user example, caution, additional reference) (see Figure 13-11).

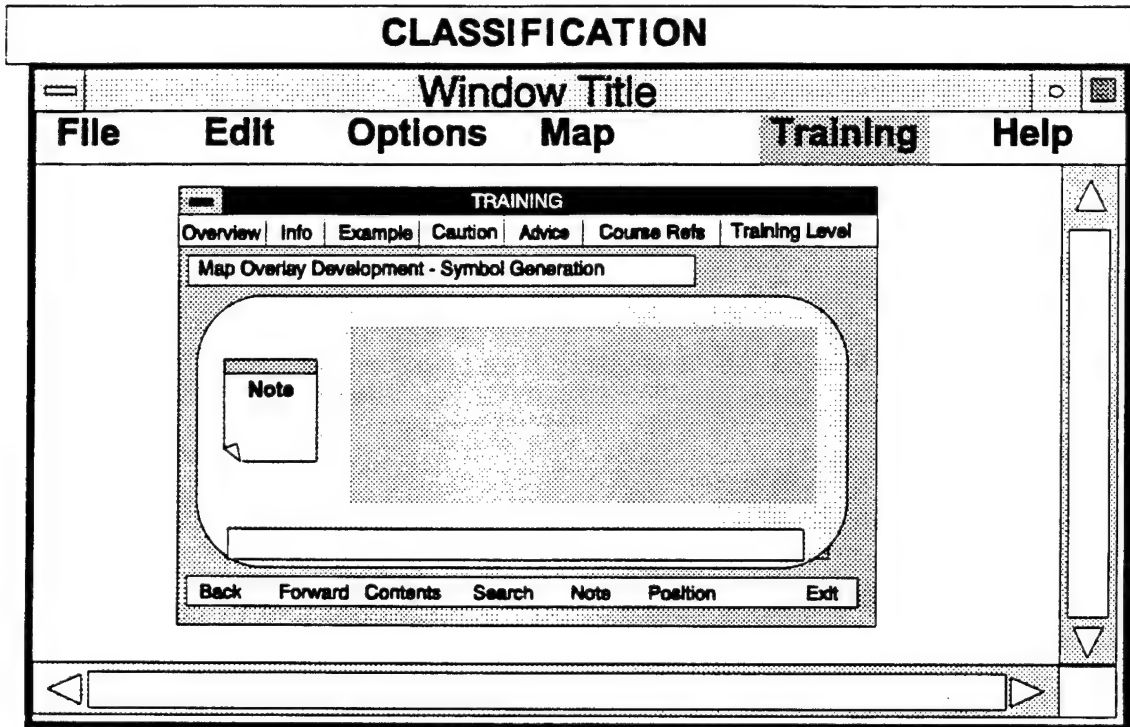


Figure 13-11. Example of Identification Annotation Position

This page intentionally left blank.

## REFERENCES

Paragraph	Reference
13.1.1	Nicol (1990) p. 115
13.1.2	Kearsley (1988) p. 27; Shneiderman (1987) pp. 371-372; Carroll and Mazur (1986) p. 38
13.1.3	Shneiderman (1987) p. 374
13.1.4	Kearsley (1988) p. 99
13.1.5	Gery (1991) p. 59
13.1.6	Kearsley (1988) p. 22
13.1.7	Wexelblat (1989) p. 74
13.1.8	Helander (1990) p. 603
13.1.9	Roth et al. (1988) p. 915
13.1.10	Getler (1991) pp.14-22
13.1.11	Roth et al. (1988) p. 98; Carroll and McKendree (1987) p. 24
13.1.12	Avery (1992) Personal Communication
13.1.13	Knerr (1992) Personal Communication
13.1.14	Kearsley (1988) p. 77
13.1.15	Fernandes and Maracle (1991) p. 9; Ripley (1989) pp. 811-822
13.1.16	Roth et al. (1988) p.98
13.2.1	Kearsley (1988) p. 66, 81; Wexelblat (1989) p. 76
13.2.1a	Helander (1990) p. 353
13.2.1b	Kearsley (1988) p. 66, 81
13.2.1c	Kearsley (1988) p. 66, 81
13.2.2	Carroll and McKendree (1987) p. 23
13.3.1	Sellen and Nichol (1990) p. 146; Kearsely (1988) p. 77
13.3.2	Gery (1991) p. 59
13.3.3	Gery (1991) p. 49
13.3.4	Gery (1991) p. 59
13.3.4a	Wexelblat (1989) p. 76

## REFERENCES (cont'd)

Paragraph	Reference
13.3.4b	Carroll and McKendree (1987) p. 25
13.3.4c	Carroll and McKendree (1987) p. 25
13.3.4d	Carroll and Mazur (1986) p. 39
13.3.5	Gery (1991) p. 59; Wenger (1987) p. 124
13.3.6	Gery (1991) p. 137
13.3.7a	Seybold's Office Computing Report (1989) p. 6
13.3.7b	Wexelblat (1989) p. 76
13.3.7c	Wexelblat (1989) p. 76
13.4.1	Wenger (1987) p. 336
13.4.2	Fernandes and Maracle (1991) p. 9
13.4.3	Fernandes and Maracle (1991) p. 9; Seybold's Office Computing Report (1989) p. 6
13.5.1	Badler and Webber (1991) p.71; (1990) p. 638
13.5.2	Helander (1990) p. 358
13.5.3	Nicol (1990) p. 115; Roth et al. (1988) p. 118; Carroll and Mazur (1986) p. 38; MIL-STD-1379D (12.5.1990) p. C-5
13.5.4	Carroll and Mazur (1986) p. 37; Walker (1987) pp. 238-243
13.5.5	Kearsley (1988) p. 24
13.5.6	Roth et al. (1988) p. 30
13.5.7	Wexelblat (1989) p. 74
13.5.8	O'Malley et al. (1983) p. 6; Carroll (1982) pp. 49-58
13.6.1	Kearsley (1988) p. 67
13.6.2	Seybold's Office Computing Report (1989) p. 5
13.6.3	Kearsley (1988) p. 79
13.7.1	Fernandes and Maracle (1991) p. 9
13.7.2	Brooks et al. (1990) p. 1387
13.7.3	Fernandes and Maracle (1991) p. 9
13.7.4	Kearsley (1988) p. 76

## REFERENCES (cont'd)

Paragraph	Reference
13.8	Sellen and Nicol (1990) p. 145; Helander (1990) p. 360
13.9.1	Fernandes and Maracle (1991) p. 10
13.9.2	Fernandes and Maracle (1991) p. 9
13.10.1	Roth et al. (1988) p. 36
13.10.1a	Roth et al. (1988) p. 36
13.10.1b	Wenger (1987) pp. 296-297
13.10.2	Roth et al. (1988) p. 36
13.10.3	Roth et al. (1988) p. 36
13.10.4	MIL-STD-1472D (1981) p. 274; Shneiderman (1987) p. 317; Smith and Mosier (1986) p. 391
13.10.5a	Wexelblat (1989) p. 76
13.10.5b	Wexelblat (1989) p. 76
13.10.5c	Carroll and McKendree (1987) p. 24
13.10.5d	Carroll and McKendree (1987) p. 24
13.10.5e	Kearsley (1988) pp. 20, 80
13.10.5f	Carroll and Mazur (1986) p. 40
13.10.5g	Carroll and McKendree (1987) p. 24
13.11.1	Kearsley (1988) p. 23
13.11.2	Kearsley (1988) p. 23
13.11.3	Wenger (1987) p. 319
13.11.4	Wenger (1987) p. 319
13.11.5	Gery (1991) p. 63

This page intentionally left blank.

## 14.0 EMERGING TECHNOLOGY

This section of the *Style Guide* is planned to provide the developer with an overview of new issues that may have an impact on the Human Computer Interface. As design guidelines for these emerging technologies mature, the information presented here may become part of an existing section of the *Style Guide* or form the base for a completely new section in a future edition. The material may be dropped from future versions if no longer relevant. This material includes discussions on personal layers and multimedia technology.

### 14.1 PERSONAL LAYER

The concept that certain computer interfaces should accommodate different user preferences is widely accepted within the software development community. In a number of situations, this may not be advisable. These situations include multi-user shared workstations, workstations used for over-the-shoulder viewing, and systems that are primarily composed of novice users. In the past and to a certain extent today, the common practice was to assume that there is one "stereotype" user group and to design the interface for that group. Stereotypes (or homogeneous groups) can be defined as user groups formed by individuals with similar or identical characteristics, needs, preferences, and capabilities.

In reality, seemingly homogeneous groups are composed of individuals with widely varying degrees of competence, preferences, and aptitudes. Therefore, in many respects, the system design did not necessarily accommodate these within-group individual differences, with subsequent performance degradation due to:

- Level of experience
- Personality traits
- Demographic characteristics
- Physiological attributes.

Designing a more personalized system that many can use and that remains responsive to individual needs is an elusive goal, primarily because computer-user populations are not homogeneous. Considering individual user differences, it may not be appropriate to design a single static interface. One approach to a personalized interface has been to design different interfaces for different groups of users. The approach requires careful examination of the user population in order to identify different user groups. This may even require different versions of the same product.

Although the need for personalizing the computer interface is generally recognized, the way to accomplish this has not been unanimously accepted. The primary methods or procedures for personalizing an interface include:



- Prototype the application in conjunction with the user.
- Allow the end user to directly modify the working environment.
- Use adaptive modeling.

The first method involves a process by which system designers consult representatives of the end-user population and develop a prototype version of the application. Potential users evaluate general functionality and appearance and comment on system quality. End-user comments are reviewed by system designers and adjustments made to the application. This iterative process continues until the user interface is complete. The end-users involved are assumed to represent the user population as a whole. This is the most common method used by the military operational community. However, a major weakness of this method is the assumption that the combination of individual users constitutes a homogeneous user population.

The second personalization method is to allow the end user to directly modify the working environment. Examples are typically found in the UNIX operating system (discussion follows on UNIX implementations of user preferences). Many researchers agree that the end user should have some ability to modify the interface. For efficiency, techniques are included to allow the experienced user to speed up interactions in natural ways (e.g., enter different information items in the same line to avoid the need for individual prompts [prompt-suppression]). Disadvantages of direct user modification include:

- Difficulty for casual users learning to make modifications
- Having to trade-off between setup time and task to be accomplished
- Difficulty associated with supervisor over-the-shoulder viewing
- Potential between-user difficulties.

The third method of personalization is the adaptive modeling method. Adaptive modeling describes the computer's ability to alter the interface in order to meet the changing needs of the user or to recognize users and adjust based on past preferences or behavior/activities. The system monitors user activity and tries to adapt automatically to the changing behavior.

A "user modeler" is often used to incorporate the user characteristics with other factors affecting performance, preferences, and needs of the user. The purpose of the user modeler is to predict the preferences and the current situation of the user. The user modeler receives data on the user's activities, uses this information together with the profile of the user and a knowledge base already stored on the computer and updates the user model. The model is then used to determine an appropriate interface that fits the user's characteristics, needs, and preferences. The system adapts itself and improves the model as information is collected about the user during the actual interactions.

Some adaptive applications recognize differences between novice and expert users. These interfaces may provide automatic assistance to the novice. However, the expert receives assistance only when it is requested. These applications allow the novice to learn the application more efficiently and to slowly eliminate the tutorial function as application skills improve.

Workgroup situations, such as military tactical operations or business offices, and/or network capabilities in today's workgroups introduce other issues when dealing with personal preferences. For, although it is important that the user have the capability to modify the environment, some order and limitations are necessary. The extent of these limitations depends on their impact. In addition, the challenges of designing groupware -- Computer Supported Cooperative Work (CSCW) -- add a new dimension to the role of interface designer.

Two examples of implementing user preference files can be found in the UNIX operating system and in the WinLogin feature available for the Microsoft Windows operating system. A general outline of each is discussed in the following paragraphs.

Because UNIX is a multiuser system, files can be created by individual users who "own" them until they are deleted or given to another user. Adding a new user to the system requires a user name, group, login identification (ID), and password. Each user belongs to a group and can share files with other members of the group. When a file is created, the user and group are automatically given permission to access the file. The user can add permissions for others or take away the group permission.

In the UNIX system, the user's login ID must be unique to the system. Each user is assigned a "home" directory, which contains the user's personal files. Included in the home directory are the "dot files." The types of preferences specified through dot files include capabilities to:

- Store environment variables
- Store commands that would be typed at the command line
- Create aliases (shorthand forms) for commonly used commands
- Start a window manager (e.g., Motif, Open Look)
- Specify/modify the interface appearance in terms of colors and fonts
- Specify menu items and mouse buttons for selection
- Specify tools and applications available.

The UNIX environment provides the opportunity to customize the working environment, but not without drawbacks. When loading any software applications, environment variables and paths must be set up according to proper specifications. When user environment variables conflict or overwrite software environment variables, the software may not run without making changes. This can make software installations difficult and require the services of system administrators.

Microsoft WinLogin provides a tool for managing workstations on a network running the Microsoft Windows operating system. A user's windows environment is defined by settings in various files (.INI, .PIF, and .BAT) called configuration files. Windows configuration files and configuration files for Windows-based applications are placed at a central location on the network. The network administrator manages the whole set of configuration files as upgrades are made and new applications are added.

A database keeps track of the locations where all configuration files are stored. The network administrator can modify the database using a Configuration Manager. This method facilitates setting up a new application or changing characteristics for groups of users or types of workstations by changing a single configuration file.

WinLogin enables users to log on to any workstation and see their own customized Windows environment. When the user logs in and starts Windows, WinLogin checks the database to locate the user's files, the files for the workstation, and the default settings. These settings are combined based on the merge rules for each database. These merge rules specify that particular entries come from the administrative settings, while others come from workstation and group settings. The merge rules also specify whether "supervisory" entries can be replaced by user preferences.

The following guidelines apply to personalization of the user interface.

#### **14.1.1 Levels of Expertise**

- Examine the user group to determine the needs of the individual end user. As a minimum, include expectations of the user's level of experience, personality traits, and demographic and physiological characteristics.
- Provide for the user who is experienced on command line interfaces by allowing for the use of both computer menu sequences and direct commands.
- Provide an adaptive interface design and the capability for the application to interact with the end users on their level of proficiency.
- Determine the standardization requirements of the group (business or operational) before allowing user-controlled interface modification.

#### **14.1.2 Experienced Users**

- It is recommended that applications provide program shortcuts for experienced users.
- If direct commands are offered, they should provide a more efficient selection method when proficiency is attained.

- Provide a facility for user-defined abbreviations or aliases (e.g., alias *bye* for *exit* or *logout*).
- Permit the user to specify characteristics of the help system.

### **14.1.3 Novice Users**

Provide the novice user with information and direction. Lead the novice through a solution separate from error messages, which allows the user to call up additional detail. Allow the novice end user minimum options to alter the computer system interface, while allowing the novice user to develop familiarity by using the menu-driven sequences.

### **14.1.4 Interaction Styles**

Design user interfaces uniquely (because individuals are unique) with regard to distinct needs and differences for greater effectiveness. Incorporate the following:

- Ensure that the system is adaptable to the physical, emotional, intellectual, and mental traits of the end-user population.
- Ensure that the system responds to individual differences in interaction manner, depth, and style.
- Utilize user 'stereotypes' in constructing an effective model until preferences are identified.

### **14.1.5 Interface Personalization**

The following principles apply to interface personalization. However, it should be noted that there are situations where personalization is not recommended.

#### **14.1.5.1 Workstations**

- Design keeping in mind that differences among users have a greater impact on performance level than differences in system designs and training methods.
- Improve user productivity and efficiency by improving the system's ability to adapt to various user preferences.
- Ensure that the user takes only a minimal amount of time to personalize an interface. If personalizing a system is too complex or requires a considerable amount of time, the effort to personalize will not be cost-effective.
- Enable users to change the appearance of the system interface by changing colors and fonts on the screen, except in circumstances where color is required to be fixed (e.g., security or classification coding).

- Allow users to modify the locations of windows and tool bars.
- Provide options for user manipulation that enable the user to tailor the terminal to his or her individual needs.

#### **14.1.5.2 Networks**

- Ensure that network systems allow the user to work in a personalized atmosphere (i.e., users should have the same flexibility that a stand-alone system would offer). To accomplish this, allow each user his or her own network account.
- Allow the user to change the interface so that the same terminal can be attached to different host systems (e.g., make the terminal into a network node by setting the appropriate communications parameters and loading suitable emulators).

#### **14.1.5.3 Messages**

- Allow the user to express the same message in more than one way.
- Provide the user with every opportunity to correct his or her own errors.

### **14.2 MULTIMEDIA**

Multimedia blends publishing, entertainment, and computers into a medium for information exchange that expands the potential for all three. Building a multimedia application, often referred to as a "title," requires a mixture of expertise including programmers, writers, artists, musicians, and sound engineers, as well as a multimedia producer to coordinate the activities of the team.

Multimedia elements (e.g., sound, video, animation) are typically sewn together using authoring tools. These software tools are designed to manage multimedia elements and provide user interaction. Interactivity is a main ingredient of multimedia. The user is in control -- what the user sees and hears is the result of choices and decisions the user has made. An important requirement for multimedia applications is that the designer create an interactive environment in which the user is in control and the user is comfortable being in control.

Most authoring tools also offer facilities for creating and editing text and images, and extensions to drive video players, videotape players, and other relevant hardware peripherals. Sound and movies are usually created with editing tools dedicated to these media, and are then imported into the authoring system for playback.

The sum of what gets played back is the human computer interface, and this interface rules both what happens to the user's input and the actual graphics on the screen. Unlike the linear sequence, which defines the order in which the pages of a book are read, a multimedia application allows the user to shift the information focus in a nonsequential manner depending on the reader's interests. The structure of the applications provides options for the reader (e.g.,

Go to B, C, or D). The author of the text can set up a number of alternatives for readers to explore rather than a single stream of information.

The strengths of multimedia arise from the flexibility in storing and retrieving knowledge. Any information, be it text, graphics, sound, or numerical data, can be linked to any other piece of data, making it possible to create a "seamless information environment."

The hardware and software that govern the limits of what can happen are the multimedia platform and environment. The paragraphs that follow will expand on the various elements that make up the hardware and environment and some emerging guidelines for their use and specification.

Two terms (hypertext and hypermedia) require explanation prior to proceeding.

- **Hypertext** - essentially the ability to link specific text to related text, or in some cases, visual elements. The words, sections, and thoughts are linked together and can be navigated in a nonlinear fashion. Hypertext is an extremely powerful information tool because it allows the representation of knowledge, browsing, carrying out structured searches, and making inferences, all within the same environment.
- **Hypermedia** - when associated images, video clips, sounds, and other exhibits are added to the hypertext. Also, when interaction and cross-linking are added to multimedia and the navigation system is nonlinear, multimedia becomes hypermedia.

Most of the information in the following paragraphs is true of hypertext and hypermedia. These two terms are often used interchangeably in the literature.

#### **14.2.1 Multimedia Personal Computer (MPC)**

The Multimedia Personal Computer (MPC) Marketing Council's MPC specification requires that machines bearing the "MPC" trademark offer a core set of features. The specification has been divided into Level 1 and Level 2. The Level 2 specification is recommended. It is also recommended that the hardware acquired be the most affordable in each feature category. Make sure the system allows room for expansion.

A number of companies offer fully integrated Multimedia PC hardware systems. Alternately, upgrade kits are available to transform a current PC hardware (80286 up) into a Multimedia PC. The kits usually contain a sound card with Musical Instrument Digital Interface (MIDI), Compact Disk-Read Only Memory (CD-ROM) drive, and Multimedia Windows software.

##### **14.2.1.1 CPU**

Hardware equipped with an 80286 or compatible processor chip is required for Level 1. Generally, at least a 80386 or 80486SX processor or equivalent is recommended; the highest affordable clock speed is desirable. The Level 2 specification requires a 486SX-25.

#### **14.2.1.2 RAM (Random Access Memory)**

A minimum of 2 megabytes (MB) RAM is required by MPC Level 1 specification and 4 MB for Level 2. At least 8 MB is recommended for authoring and at least 6 MB for a system that will be used as a presenter only. All systems potentially benefit from more RAM.

#### **14.2.1.3 Magnetic Storage**

- A 3.5" floppy drive with 1.44 MB capacity is required.
- A hard drive of at least 30 MB is required for Level 1 and 160 MB for Level 2, but 300-600 MB is recommended.
- A tape drive and tape backup are recommended.

#### **14.2.1.4 Optical Storage**

CD-ROM, with compact disc (CD) digital audio output and data transfer rate of 150 kilobytes (kB) per second, is required for the Level 1 specification MPC standard, but 300 kB is required for Level 2. The fastest transfer rate available (at an affordable cost) is recommended.

#### **14.2.1.5 Audio**

The following audio hardware is required for MPC:

- An 8-bit for Level 1 and 16-bit for Level 2 digital-to-analog converter (DAC) 22.05 and 11.025 kilohertz (kHz) rate
- An 8-bit for Level 1 and 16-bit for Level 2 analog-to-digital converter (ADC) 11.05 kHz rate, microphone level input
- A music synthesizer capable of four to nine instrument synthesis
- An on-board analog audio mixing capability.

#### **14.2.1.6 Video**

VGA (16 colors) color graphics adapter is required, but Super Video Graphic Adapter (SVGA) (256 colors) is recommended. This is still inadequate for producing realistic images or video.

Higher than 640 x 480 resolution is only important when the display is larger than 14". On a 14" display at a normal desktop viewing distance of about 18", the user cannot distinguish more than 640 pixels across. When using a 16" or 19" display at this viewing distance, an Extended Graphic Adapter (EGA) display of 1024 x 768 can be beneficial.



#### **14.2.1.7 I/O Hardware**

The following I/O hardware is required for MPC:

- An 101-key keyboard
- A two-button mouse (three-button is acceptable)
- A MIDI I/O port
- A serial port
- A parallel port
- A joystick port.

#### **14.2.2 Audio**

Audio is an integral part of the multimedia environment adding the dimensions of speech, music, and/or sound effects. The ability to capture natural sounds and bring them into a multimedia application is the purpose of digitized audio. This involves more than setting up a microphone and telling the computer to capture the sound. A well-produced sound track is the best means to enhance the realism and effectiveness of the application interface. The following paragraphs discuss the facilities necessary to capture and process audio.

##### **14.2.2.1 Audio Digitizers**

Audio digitizers are devices for recording and playing back digital audio. They range from the simple add-in sound cards to the audio production systems costing thousands of dollars. The principal technical descriptive characteristics consist of sampling rate, sampling size, and resolution. The selection decision should be based on a trade-off analysis of user functional needs and equipment availability.

- **Sampling Rate:** The sampling rate is like the frame rate at which film or video is played back. Sounds sampling or digitizing captures "snapshots," samples of sounds that are played back rapidly. The MPC specification requires mono playback at 22.05 kHz but recording at a rate of 11.025 kHz. Most PC sound boards allow playback and recording at 44.1 kHz (compact disc audio plays back at a rate of 44.1 kHz).
- **Sample Size:** The amount of information stored about each sample. Sample sizes are typically either 8-bit or 16-bit. The larger the sample size, the better the data describes the recorded sound. While 8-bit sound provides 256 units to describe dynamic range and amplitude, 16-bit provides 65,536 units.



- **Resolution:** The resolution is the number of bits used to represent an individual sample. The more frequent the sample and the more data stored about the sample, the finer the resolution and quality of the captured sound when it is played back. It is analogous to the number of bits used to represent a pixel on a screen. As an 8-bit image is grainier than a 16-bit image, an 8-bit sound is grainier than a 16-bit sound. The MPC standard supports the 8-bit rate, but the 16-bit rate is recommended for quality sound reproduction.

#### 14.2.2.2 Sound Editors

Sound editors (also called sample editors) provide tools to record, edit, rearrange, mix, process, and playback sound files in a variety of formats. Sound is represented as an amplitude waveform, with time corresponding to the horizontal axis and sample value (i.e., volume or intensity) assigned to the vertical axis. Within this format, the user can zoom out to view an entire sound file or zoom in as close as a single sample. This allows cutting, copying, and pasting of sound data with a precision of up to 1/44,100 of a second, depending on the digital signal processor (DSP) board and software.

Sound editors are available both as stand-alone products and bundled with digital audio cards. Most bundled editors include a limited number of features such as cut, paste, fade-in and out, and amplitude adjustment.

Full-fledged editors provide signal processing options such as cross-fading, which allows one track to fade in while another fades out; digital equalization features, such as boosting or cutting the volume of selected frequencies or frequency bands; time compression and expansion, increasing or decreasing the length of a sound file region without changing its pitch; and pitch shifting, changing the key of a passage without altering its duration. These options are invaluable for fitting a piece of audio to video not specifically created for the sound or sound not created for video.

#### 14.2.2.3 MIDI

MIDI is an international specification used by electronic musical instruments to communicate with each other, computers, mixers, and other devices. MIDI specifies the cabling, hardware to connect MIDI devices, as well as protocol to communicate between these devices. Any musical instrument with a microprocessor to process MIDI messages can be a MIDI device.

Whereas digital audio actually records and stores the sound, MIDI simply describes the performance. MIDI describes what notes are being played, when they are played, and with what nuance (e.g., sustain, pitchbends, vibrato). Since MIDI deals only with events that trigger sound, the files are rather small (e.g., a one-minute sampled composition requires 12 MB to store but only 15 kB MIDI file). Playing back a MIDI file requires a musical instrument (which could consist of a box and speaker outputs), while the sound recording requires only an amplifier and speakers.

- IBM- and MPC-compliant sound cards usually include basic MIDI interfaces.

- Dedicated interface products are more appropriate for musicians and multimedia experts. They often allow increasing the available channels and therefore allow controlling more devices.

#### **14.2.2.4 MIDI Sequencers**

MIDI sequencers record and store musical events played on a MIDI instrument such as a synthesizer or sampler. They do not record the sound but record the MIDI data describing the sounds. Most MIDI sequencing software mimics a typical multitrack tape recorder and offers standard editing features such as cut, copy, paste, merge, insert, pitch correction, transposition, inversion, retrograde, tempo changes, and score edits.

A MIDI file can contain up to 16 channels of music data, allowing recording of and playing back of many different musical instruments, each on a different channel. The general MIDI numbering system from 0 to 127 identifies instruments that can be synthesized, although MIDI is flexible enough to allow remapping to non-standard instruments. MIDI also allows the user to set up any instrument to receive on any channel (e.g., data could be received from a keyboard synthesizer and played back on a another keyboard synthesizer or MIDI instrument).

#### **14.2.3 Images**

Software is available to support nearly every combination of 3-D modeling, lighting, defining surface attributes, animating, and rendering. Selecting the appropriate software depends on the type of graphics. Print work, animation, visualization, fly-throughs, slides, and multimedia all require different subsets of features.

The computer creates still images in two ways -- as bitmaps (or paint graphics) and as vector-drawn (or drawn) graphics. Bitmaps are used for photo-realistic images and for complex drawings requiring finer detail. Vector-drawn objects are used for lines, boxes, circles, polygons, and other graphic shapes that can be expressed in angles, coordinates, and distances. A drawn object can then be filled with color and patterns.

The appearance of these graphics depends on the display resolution and capabilities of the computer's graphics hardware and monitor. The images are stored in various file formats and can be translated from one application to another and from one computer platform to another. They are typically compressed to save memory and disk space.

Programs are available for converting between the two formats. Converting a drawn (or vector) object to a bitmap is far easier than the reverse.

##### **14.2.3.1 Painting Programs**

Clip Art is an example of commercially available bitmapped graphics. Clip Art can be manipulated, and properties such as brightness, contrast, color depth, hue, and size can be adjusted.

Bitmap editors are usually called paint programs. They are the closest to the traditional artist's media and creative processes. They determine the color of each pixel they touch. The tools shape and shade the images by manipulating brush type, geometry, and ink style. The greater the control, the greater the number of effects that can be created.

Fill tools place solid colors, textures, and patterns in the designated areas. Areas can also be selected for cutting, copying, and pasting operations as well as rotating and scaling.

Considerations in purchasing a painting tool include the capabilities of the toolkit, maximum image size and resolution, interface resolution, and the number of colors available.

#### **14.2.3.2 Drawing Packages**

Computer-Aided Design (CAD) programs traditionally use vector-drawn graphics for creating the highly complex and geometric renderings needed by architects and engineers. Programs for 3-D animation also use drawn graphics.

Vector graphics software are called draw programs. These tools are similar to those used in mechanical drawing. They generally include tools for creating geometric shapes, lines, and curves. Objects can be moved, scaled, rotated, copied, and attributes changed.

#### **14.2.3.3 Animation**

The entire project can be animated, or animation can be used for accenting. Animation can consist of as many as 30 images per second. One way to generate animation is to create a series of still images individually and use animation software to flip through them like a movie.

The capability to import and integrate images from a wide variety of sources is an important feature. Fast rendering, timeline- and keyboard-based animation interfaces, and 3-D fonts are features of a higher end tool.

#### **14.2.4 Video**

Video differs from animation in that video describes images of real events stored in a digital format, whereas animation is simply computer-generated images. Video image files usually contain audio tracks and are larger than animated images.

Although many applications are created using images or animation or both, video appears to have the greatest user impact. Video must be well planned to have the greatest impact and effectiveness. Integration of video into the application and disk storage space are key elements for successful video.

Two new products for video incorporation using only software support are described, followed by a discussion of hardware-based tools.

#### **14.2.4.1 QuickTime Movies**

Many Apple-Macintosh applications can create or play QuickTime movies. QuickTime for Windows (QTW) allows PC users to access all of the QuickTime movies available for the Mac. This enables the same video clips to be used on both platforms.

Dedicated editing and effects software is the best choice for polishing productions. Just about anything you can do in a broadcast-quality editing suite has a QuickTime equivalent, limited only by today's lower hardware resolution and size and frame rates. Editing functions include "log," mark, and identify scenes or picture sequences; trim to desired length; and order them for playback. Most editors have a visual interface to identify and sort clips and a timeline view for sequencing and trimming elements.

Some editors allow creation of transitions, adding titles and graphics and applying various image transformations and filters such as traditional wipes, flips, and turns as well as digital domain unique morphs and melts. The transitions and overlays available are dependent on the special effects capabilities of the system to translate perfectly.

Audio support is often limited to capturing audio on suitably equipped computers, trimming the segments and adjustment of audio and visual segments. For the full range of audio effects, a sound editor is necessary.

#### **14.2.4.2 Video for Windows**

Video for Windows (VFW) is also known as Audio Video Interleaved (AVI). It allows the developer to capture, digitize, and compress (using a number of different compression algorithms) video. Because of the software-only-compression, compromises must be made. The image size is small, and interleaving is required to synchronize audio and video.

VFW is scalable; it can be played back on the user's PC, with an additional video decompression board. The quality of the video being played back depends on the power of the playback PC. The software drops frames when necessary, to ensure that the audio stays synchronized to the video sequences.

On a slow 386, the video may play back at 10 frames per second (fps). On a fast 486, the playback can be at 24 fps. The slower 10 fps rate will produce low quality video with a large amount of flicker.

VFW is installed as a multimedia device.

#### **14.2.4.3 Video Capture**

Capturing still images from video segments is difficult; grabbing complete sequences of images truly taxes the current capabilities of the system. Uncompressed, full-motion video is impractical because 30 seconds of full-motion video stored in analog form requires over 500 MB of storage. To make video manageable, the file must be compressed.

A growing number of manufacturers are offering add-on boards, called frame-grabbers, that grab and store movie-video images in digital form. Video capture boards can be distinguished by whether they convert full-screen, full-motion video [30 frames per second in the U.S. - National Television Systems Committee (NTSC) standard, or 25 frames per second in the European Phase Alteration Line (PAL) standard] or by whether they capture selected frames or partial screens.

The number of colors varies. Most designs that started in the video world grab 16 bits per pixel, while most computer-oriented products capture 24 bits per pixel. Few boards can grab a complete video signal at full speed.

- Applications include traditional video editing, multimedia presentations, training videos, kiosks, scientific analyses, and archival storage.
- Some are offered as separate boards that accept analog video as inputs and produce digital files as output.
- More often, digitizers are combined with video capabilities in a single board in a motherboard-daughterboard configuration or as side-by-side boards connected by a ribbon cable.

#### **14.2.4.4 Compression**

Many products trade off a higher frame rate for lower pixel depth or smaller image size. Boards that capture images at rates approaching full speed can normally do so only to RAM, since data cannot be written to a hard disk fast enough.

The current generation of PCs requires hardware-assisted compression to capture full-screen, full-motion video. Most boards have relied on either Intel's Digital Video Interleaved (DVI) chip sets or on C-Cube CL- 550 Joint Photographic Experts Group (JPEG) processor. Since video compression standards have been in flux, companies have tended to use various methods. Systems based on the JPEG are the most prevalent, while DVI and Motion Picture Expert Group (MPEG) are gaining popularity. A brief discussion of these three standards is presented below.

- Under JPEG, an image is divided into 8 x 8 pixel blocks, and the resulting 64 pixels (called a search range) are mathematically described relative to the characteristic of the pixel in the top-left pixel. Since the binary description of this relationship requires less than 64 pixels, more information can be transmitted in less time. JPEG is primarily used to encode still images and compresses about 20:1 before image degradation occurs. Compression is slow. JPEG does not handle black and white (1-bit per pixel) images.
- MPEG is used to encode motion images. MPEG compresses at a 50:1 ratio before degradation of the image occurs. Ratios of 200:1 are attainable, but observable degradation occurs. The compression rate is fast enough to allow CD players to play full-motion color movies at 30 frames per second.

- DVI is a proprietary, programmable compression/decompression technology. The hardware consists of two components to separate the image processing and display functions. It allows compression of video images at ratios between 80:1 and 160:1. DVI will play back video in full frame size and in full color at 30 fps. When tied in with a mainframe computer, DVI playback approaches the quality of broadcast video.

Although DVI claims to offer greater compression ratios, JPEG independently codes each frame. This allows frames to be edited or rearranged. Faster DSPs or faster CPUs may eventually provide new compression methods.

Boards without dedicated chips can compress video captured into memory. A PC or Mac with 8 MB of RAM can usually capture a few seconds of partial-frame video before stopping to compress and save to disk.

Most applications can use less costly partial-screen or slower-frame-rate videos. These applications (e.g., electronic mail or training) often limit video to partial screen in order to provide room for other program elements.

Systems designed to play back QuickTime movies on the Mac and PC or AVI on the PC are limited by the playback hardware. Most Macs and PCs are limited to a partial screen and about 15 frames per second.

#### **14.2.4.5 Video-Editing Software**

Desktop video-editing systems vary widely in capability to handle video. Less expensive packages deal mainly with control and status information, while the actual video signals are routed directly from recorder to recorder or recorder to screen. The packages are classified as follows:

- **Cuts-only** - unadorned final output is copied directly from one deck to another
- **Off-line** - an edit decision list (EDL) will be exported to a more sophisticated editing system
- **On-line** - can add graphics, transitions and special effects to video. On-line systems can also produce EDLs for further work on larger systems.

Generally, compressed video is good enough for EDL; but the amount of compression required to get the original video down to practical sizes squeezes out a fair amount of the picture quality. Most digital systems are used as off-line feeders to an on-line system. However, as compression technology advances and storage options become less expensive, more digital systems will offer direct output alternatives.

Older desktop video editing systems are similar to traditional editing systems. The newer systems have graphical interfaces using a point-and-click operation. The intended audience

often dictates the capabilities required. The ability to connect to video decks and recorders through distributed control networks allows editing of specific frames.

#### **14.2.5 Text**

Many multimedia applications are primarily text-driven. Text-based files form one of the largest sources of information. Often multimedia projects are developed by converting a book into an on-line application. The three main ways to get text into compatible forms include the following.

##### **14.2.5.1 Retyping**

Although retyping the text can be labor-intensive, it is often the most economic way to import large amounts of printed material.

##### **14.2.5.2 Scanning**

Scanning can be an efficient way to get text into a computer. The scanner converts pages of text into bitmapped images. Software is used to analyze the letter shapes and convert them to ASCII letters. Utility programs are available to detect misreads and scanner errors.

##### **14.2.5.3 Computer-Supported Conversions**

Converting involves transferring electronic files between different formats. Converting text always results in the loss of some original formatting. Proper formatting, indexing, and other reference tags are required to make the text useful. Suggested formats include straight American Standard Code for Information Interchange (ASCII) text, Rich Text Format (RTF), Standard Generalized Markup Language (SGML), and Document Control Architecture (DCA).

#### **14.2.6 Compact Disc Technology**

CD technology is gaining acceptance as an economical storage medium, which is ideal for delivering large programs such as reference material and multimedia titles. PCs, Macs, and workstations are now available with CD-ROM drives and upgrade kits. Drives are becoming less expensive, and consumers and education audiences have increased expectations.

##### **14.2.6.1 CD Specifications**

MPC Marketing Council has stated that CD-ROM drives must be able to read multisession recordings and be CD-ROM eXtended Architecture- (XA) ready. The XA files produced by the Kodak Photo CD format is an example of ready-to-read XA files. Competing CD formats include Commodore's Dynamic Total Vision (CD-TV), Sony-Phillips' Compact Disc - Interactive (CD-I), Tandy's Video Information System (VIS), Sony's Multimedia CD Player (MMCD), and Kodak's Photo CD.



#### **14.2.6.2 Storage Capacity**

A single CD-ROM disc can hold up to 680 MB of information. This equates to 150,000 printed pages or approximately 250 large books on one compact disc.

#### **14.2.6.3 Data Transfer Rate**

Data transfer rates have increased from 150 kB per second to 300 kB per second. Increased speed allows smoother audio and video playback. Although new 16-bit game cartridges are better, only CD-ROM has the potential to truly increase games productions, which will include lengthy stereo audio and full-motion video clips.

#### **14.2.6.4 Access Time**

The average access time is the time it takes to find what you want to read from the disc. Average access times (also called seek time) for state-of-the-art systems is about 280 ms, although most drives are still in the 350-380 ms range. The MPC standard is anything under 1000 ms, as opposed to 15-30 ms for most contemporary hard drives.

The objective in designing a multimedia interface is to reduce the number of seeks the drive makes in order to access the data. Advanced CD mastering packages allow selection of the ring where data are located.

#### **14.2.6.5 Mastering the Title**

The process of turning an application and its associated files into a CD-ROM disc includes premastering, final testing, and mastering and replication.

#### **14.2.7 Authoring Systems**

Multimedia elements are typically sewn together using authoring tools. These tools provide the basic building blocks and framework for creating a multimedia application. In designing the multimedia project, the traditional scripting and design methods (e.g., copyboard drawings and typed scripts) or software tools can be selected.

Most authoring tools can be used by non-programmers, although some programming is useful. Authoring tools are best suited for content-rich applications (e.g., those loaded with text, images, and sound) because they specialize in data delivery. Their benefits include ease of use, fast development cycles, predictable characteristics, and reliability.

The target audience is the most important factor in selecting an authoring system. Developing a package for in-house use or controlled situations (e.g., kiosks) provides more leeway than productions for a demo or a tutorial for distribution to a large audience.

Creating a presentation capable of running consistently on almost any machine requires an authoring system suited to the purpose. Multimedia desktop presentation packages vary from



pure 2-D animation programs to traditional charting packages with a few added multimedia capabilities. These evolving presentation packages may be the best choice for users already familiar with charts and slide shows.

Features of a good authoring system include:

- Capability to integrate text, still graphics, animations, sound (digitized, MIDI or CD-Audio), and video
- A visual flowcharting system, storyboards, navigation diagrams, or overview facility for illustrating the project structure at a macro level
- Support for creating tailored presentations, either through scripting language or other means (often icon-based programming)
- Support for one or more levels of interactivity
- Capability to allow specification of timing and sequence on systems with different (faster or slower) processors
- Provision of a playback feature for building and testing segments of the project
- Permission to distribute run-time files created
- Capability to add new features or extensions.

Many of the newer packages take a middle-of-the-road approach by combining text, graphics, sound, video, and 2-D animation. Other packages contain tools to create objects and other tools to animate them. As the artistic power and complexities increase, so does the time and training required to master them.

#### 14.2.7.1 Types of Authoring Tools

Various authoring tools can be grouped based on the concepts used to sequence and organize multimedia elements and events. In choosing an authoring tool, the developer must ensure the tool supports the types of things the application will do. The various groupings are discussed below:

- **Card- or Page-Based Tools** - Elements are organized as pages of a book or stack of cards. The pages or cards can be linked in an organized sequence. The user can jump, on command, to any card in the sequence.
- **Icon-Based Tools** - Multimedia elements and interaction cues (events) are organized as objects in a structural framework or process. They simplify project organization and display flow diagrams of activities along branching paths.

- **Time-Based Tools** - Elements and events are organized along a time line, with resolutions as high as 1/30 second. They are best when the project's message has a beginning and an end. The sequentially organized frames are played back at the speed set and other elements triggered at a given time or location. More powerful tools allow jumps to any location in the sequence. These jumps provide navigation and interactive control.

#### 14.2.7.2 Multimedia Integration Tools

Multimedia brings together data from a variety of sources. The authoring package should allow importing files created in a variety of formats, including graphics, animations, sound, and text. Text should be capable of being imported with format intact (see Paragraph 14.2.5). Software should allow integrating graphics files created in other programs with minimal editing or use of specialized conversion packages.

Multimedia integration tools fall somewhere between presentation and animation packages. They have a lot in common with authoring packages but do not usually include the scripting languages of authoring systems. Some features include:

- More control of the various media than with presentation packages
- Allow capture or import graphics and text from other programs
- Provide the ability to put object in motion and synchronize that motion to sound and video clips
- Provide support for multimedia peripherals.

#### 14.2.7.3 Platform

Platform considerations for the authoring systems include the following:

- **Personal Computers** - If the authoring system runs in Windows, although this will assure the presentation runs in Windows, remember that there are several types of Windows. While many features of Multimedia Extensions for Windows are built into Windows 3.1, some are not. A DOS-based product will provide access to the widest possible audience. Ensure that the system supports the wide range of video adapters, memory- addressing schemes, and other features available on DOS machines. Note that not all DOS machines have a mouse, but MPC-compliant machines will.
- **Macintosh** - Ensure that the platform matches the color and software requirements (e.g., System 7 or QuickTime) specified by the authoring package. Requiring such devices as CD-ROM drives, laserdisc players, or MIDI interfaces may require separate routines.
- **UNIX** - Few commercial tools are currently available.

- **Cost** - Prices for authoring systems vary widely. Learning curves and technical support should also be considered when deciding on an authoring system. Some vendors include training as part of the product package.

Consider also the costs of distributing presentations (e.g., runtime versions and unlimited distribution licenses). Some include no-cost runtime licenses, while others adjust their price according to the number to copies you want to distribute.

#### **14.2.8 Design Guidelines**

The user interface is the portion of the multimedia application that presents the choices and requests to the user, receives input from the user, and provides feedback about status. The designer should not assume any knowledge on the part of the user. All information about what to do next should be constantly available on screen or in audio.

##### **14.2.8.1 User Task Analysis**

- The multimedia interface designer must have a full appreciation of how the user will use the application and what is expected from the application.
- Develop a road map of how the user will proceed through the application.
- Use an expert on the textual content for determining what topics need to be related, how available information should be divided into digestible topics, and the order in which the topics should be presented.

##### **14.2.8.2 Novice vs. Expert Interface**

- Provide plenty of navigation power, access to content and tasks for users at all levels, and a HELP system for reassurance. A separate interface for users at different experience levels is not necessarily the best method in multimedia projects.
- Present all information in easy-to-understand structures and concepts; use clear textual clues.
- The best user interface requires the least learning effort.

##### **14.2.8.3 Design Consistency**

- Ensure consistent internal design (e.g., topic screens look alike, type faces are consistently used).
- Integrate all elements, such as audio, graphics, and animation, cleanly into the overall feel of the application.
- Take time to identify some basic design standards at the beginning of the design process.

- Decide on the overall concept or metaphor and ensure all elements build on and reinforce the metaphor.
- Ensure consistent style in terms of content and breadth of information.

#### **14.2.8.4 Data Types**

- A good multimedia application will include only a fraction of the many data types that are possible.
- Use visual tools to enhance information retrieval. Do not confuse or distract the user with visual elements used simply because they are available.
- Limiting the data types also simplifies the installation of applications on different systems.

#### **14.2.8.5 Navigation**

- An important requirement for multimedia applications is that the designer create an interactive environment in which the user is in control and the user is comfortable being in control.
- The application should allow users to start when they want, stop when they want, retrace their steps when they want to backup, and, most important, never do something they don't expect.
- Provide the user the sense of freedom of choice, but remember too much freedom can be disconcerting and users may get lost.
- Provide an escape path for the user in every part of the application. The controls for the escape should be on-screen.
- Try to keep messages and content organized along a steady stream of major subjects while allowing the user to branch outward to explore details.
- The structure can be designed as a linear sequence of chronological events but also allow jumping to a specific event.

#### **14.2.8.6 Cross-Reference Jumps**

The most common multimedia behavior is a response to clicking on active words in the client area of the application. Clicking on certain text strings causes a new linked screen of information to appear. These text strings are referred to as "hotwords" or "hotspots." The result of clicking on a hotword is a jump. Picture hotspots work in the same manner. Most authoring tools provide a means to connect related topics through cross-reference jumps.

Buttons, hotwords, and picture hotspots are the primary means of control in multimedia applications. When designing cross-reference jumps, remember:

- Large numbers of jumps make navigation complex and increase test time.
- Ensure all jumps serve a useful purpose.
- Jumps should only occur between directly related or equivalent topics.

Provide an easy way out of the side trip. The user should never feel penalized for exploring (e.g., escape path discussed in Paragraph 14.2.8.5d).

## REFERENCES

Paragraph	References
14.1	Aykin and Aykin (1991); Egan (1988); Elkerton and Williges (1989); Fowler, Macaulay, and Fowler (1985); Greenberg and Witten (1985); Hansen (1971); Innocent (1982); Rich (1983); Shneiderman (1982)
14.1.1a	Egan (1988) p. 543; Aykin and Aykin (1991) pp. 373-374
14.1.1b	Elkerton and Williges (1989)
14.1.1c	Greenberg and Witten (1985) pp. 31-33
14.1.1d	Aykin and Aykin (1991) p. 373
14.1.3	Greenberg and Witten (1985) p. 32; Elkerton and Williges (1989)
14.1.4	Aykin and Aykin (1991) pp. 373-374
14.1.5.1a	Egan (1988) p. 543
14.1.5.1b	Greenberg and Witten (1985) p. 31-33
14.1.5.1c	Greenberg and Witten (1985) p. 31-33; Elkerton and Williges (1989)
14.2	Bunnell (1993); Microsoft (1991b); Vaughan (1993); Rosenberg (1993); Burger (1993); Luther (1992); Wodaski (1992); Nielsen (1990); Shneiderman and Kearsley (1989); Seyer (1991)
14.2.1	Bunnell (1993) p. 51; Microsoft (1991b) p. 1-3; Flynn (1993) p. 30
14.2.1.1	Bunnell (1993) p. 51; Microsoft (1991b) p. 1-3; Vaughan (1993) p. 169
14.2.1.2	Bunnell (1993) p. 51; Microsoft (1991b) p. 1-3; Vaughan (1993) p. 171
14.2.1.3	Bunnell (1993) p. 51; Microsoft (1991b) p. 1-3
14.2.1.4	Bunnell (1993) p. 51; Microsoft (1991b) p. 1-3; Vaughan (1993) p. 188
14.2.1.5	Bunnell (1993) p. 51; Microsoft (1991b) p. 1-3
14.2.1.6	Bunnell (1993) p. 51; Microsoft (1991b) p. 1-3; Rosenberg (1993) p. 420-422; Burger (1993) p. 164-172; Luther (1992) p. 19-20
14.2.1.7	Bunnell (1993) p. 51; Microsoft (1991b) p. 1-3

## REFERENCES (cont'd)

### Paragraph

### References

- |          |   |
|----------|---|
| 14.2.2   | Luther (1992) p. 163  |
| 14.2.2.1 | Bunnell (1993) p. 5; Burger (1993) p. 35-38; Wodaski (1992) p. 59-60; Rosenborg (1993) p. 412-413; Vaughan (1993) p. 41-42                    |
| 14.2.2.2 | Bunnell (1993) p. 9   |
| 14.2.2.3 | Bunnell (1993) p. 14  |
| 14.2.2.4 | Vaughan (1993) p. 44-51   |
| 14.2.3   | Bunnell (1993) p. 57; Vaughan (1993) p. 60-68; 74-77; Rosenborg (1993) p. 39-41; 425; Burger (1993) p. 174-178                                |
| 14.2.4   | Rosenborg (1993) p. 44-47   |
| 14.2.4.1 | Bunnell (1993) p. 71; Vaughan (1993) p. 279-282   |
| 14.2.4.2 | Vaughan (1993) p. 279-282; Rosenborg (1993) p. 46; Wodaski (1992) p. 178-179  |
| 14.2.4.3 | Bunnell (1993) p. 73  |
| 14.2.4.4 | Bunnell (1993) p. 73; Vaughan (1993) p. 414-416   |
| 14.2.4.5 | Bunnell (1993) p. 77  |
| 14.2.5   | Microsoft (1991b) p. 8-1 through 8-11   |
| 14.2.6   | Bunnell (1993) p. 39; Wodaski (1992) p. 171; Microsoft (1991b) p. 11-3 and pp. 4-4 through 4-5; Vaughan (1993) p. 424-425; Flynn (1993) p. 30 |
| 14.2.7   | Bunnell (1993) p. 27; Vaughan (1993) pp. 5, 141, 217, 220, and 222; Microsoft (1991b) pp. 2-6, 2-12   |
| 14.2.7.1 | Vaughan (1993) p. 218-219   |
| 14.2.7.2 | Bunnell (1993) p. 27  |
| 14.2.7.3 | Bunnell (1993) p. 27  |
| 14.2.8   | Luther (1992) pp. 108-109   |
| 14.2.8.1 | Luther (1992) p. 109; Microsoft (1991b) pp. 2-7 to 2-9; Rosenborg (1993) p. 383   |
| 14.2.8.2 | Vaughan (1993) pp. 138-139  |
| 14.2.8.3 | Luther (1992) p. 110; Microsoft (1991b) p. 2-9  |

## REFERENCES (cont'd)

Paragraph	References
14.2.8.4	Rosenborg (1993) pp. 390-391
14.2.8.5	Luther (1992) pp. 11-12; Vaughan (1993) pp. 141-143; Rosenborg pp. 24-27 and 240-242
14.2.8.6	Microsoft (1991b) p. 2-8; Rosenborg (1993) pp. 240-242



This page intentionally left blank.

## **APPENDIX A**

### **SECURITY PRESENTATION GUIDELINES**

This page intentionally left blank.

## APPENDIX A

### SECURITY PRESENTATION GUIDELINES

This appendix seeks to provide a uniform HCI across all CMW applications.

The security portion of the HCI will comply with DDS-2600-6216-89 and the DIA Style Guide, from which this appendix is derived. These documents outline security-related interface requirements for workstations operating in the System High or Compartmented Mode. To ensure consistency, however, any DoD workstation security label displayed by an application should conform to the labeling guidelines in this appendix.

*(NOTE: Although the figures in this appendix are only drawn in the Motif style, the security relevant information is common to both Motif and Open Look applications.)*

#### A.1 LABEL STANDARDIZATION

The following guidelines for label presentation apply to all CMWs.

##### A.1.1 Guidelines for Label Syntax

One of the primary display requirements is to use the long names of words in all labels.

###### A.1.1.1 Sensitivity Labels

- The syntax for output of sensitivity labels for all CMWs is as follows:

CLASSIFICATION COMPARTMENT COMPARTMENT COMPARTMENT....

Examples: TS A

TS A B C D

- The syntax for input of sensitivity labels for all CMWs is as follows:

CLASSIFICATION COMPARTMENT COMPARTMENT COMPARTMENT....

CLASSIFICATION/COMPARTMENT/COMPARTMENT/COMPARTMENT....

###### A.1.1.2 Information Labels

- The syntax for output of information labels to the security banner on all CMWs is as follows:

CL CW M REL C

Where CL is classification, CW is zero or more blank-separated code words, M is zero or more blank-separated non-code-word markings, and C is either a single country code or multiple country code separated by slashes. The long form of classification, code words, and

markings are used for output. The short form of classification, code words, and markings may be used otherwise (e.g., list file command). Code words, markings, and "REL" country codes are displayed in the order in which the words appear in the encodings file.

Example: TOP SECRET BRAVO1  
SECRET ALPHA1 NOFORN  
TOP SECRET B SB REL UK  
SECRET ORCON ORG X REL UK/CAN/AUS

- The syntax for input of information labels on all CMWs is as follows:

CL CW M REL C

Where CL is the classification, CW is zero or more blank-separated code words, M is zero or more blank-separated non-code-word markings, and C is either a single country code or multiple country codes separated by slashes. The short or long form of classification and markings may be used for input. When entering an information label, code words, markings, and REL" country codes may be entered in any order.

Example: TS B1  
TOP SECRET B1  
S A1 NF  
TS B1 REL UK  
TOP SECRET A SA REL UK  
S OC OX REL UK/CAN/AUS  
SECRET OC OX REL UK/CAN/AUS

#### A.1.1.3 Information and Sensitivity Labels Together

- The syntax for output of sensitivity labels and information labels together (as in the Classification Bar of each base window) is as follows:

CL CW M REL C [CL COMPARTMENT COMPARTMENT...]

Where CL is the classification, CW is zero or more blank-separated code words, M is zero or more blank-separated markings, and C is either a single country code or multiple country codes separated by slashes.

The information labels are always followed by two or more blanks, followed by the sensitivity label enclosed in square brackets. In sensitivity labels, the short form of the classification is used. In information labels, the long form of the classification is used. The long form of the code words and/or markings is used in the information label. If the information and sensitivity labels are to be output to other than the Classification or Input Information Labels (e.g., list files with labels), then the short form of classification, code words, markings, etc. may be used.

Example: TOP SECRET A B SA [TS A B]  
SECRET A B SA SB [TS A B C D]  
TOP SECRET A B ORCON ORG X REL CAN/UK [TS A B]

- At times when both labels can be input, the user must enter the left bracket to delimit the sensitivity label. The short or long form of the code word and/or markings may be used for input of both labels.

Example:      TS A B SA [TS A B]  
                  TS A1 B1 [TS A B]  
                  S A B PX LD (TS A B C D)  
                  S A B PROJECT X LIMIDIS [TS A B C D]  
                  TS A SA OC OX REL CAN/UK [TS A B C]  
                  TS A B ORCON ORG X REL CAN/UK [TS A B C]

### **A.1.2 Guidelines for Displaying Labels**

The following guidelines apply for all displaying of information labels, sensitivity labels, and clearance labels.

- Capital letters should be displayed for all classifications and words in all labels.
- Blanks should be used to separate classifications from other words in all labels, except where there are multiple words that require the same prefix or suffix, in which case the multiple words should be separated from each other with slashes.
- The long name of words should be displayed in all labels.
- The long name of classifications should be displayed in all information labels.
- The short name of classifications should be displayed in all sensitivity labels and clearances.
- The classification should be displayed first, followed by the words in the same order they appear in the encodings.
- Whenever an information label is displayed, its associated sensitivity label should also be displayed.

### **A.1.3 Guidelines For Changing Labels**

#### **A.1.3.1 Typing Interface**

The following guidelines apply to all textual interfaces that allow users to change labels:

- When typing any label or a change to any label, the user should be able to use the following interchangeably:

Upper and lower case letters

Short and long names for classifications and words

Blanks and slashes

- The following syntax should be accepted for typed changes to any label:

[+][CLASSIFICATION] [[+]-][WORD]...

where brackets denote optional entries, "+" or "-" denotes changes, and "..." denotes zero or more of the previous bracketed entry preceded by blanks. If the input starts with a classification followed by "+" or "-", the new classification should be used but the rest of the old label should be retained and modified as specified in the input.

- It should never be possible for the user to change the value of an information label above that of its associated sensitivity label without first, or concurrently, requesting that the sensitivity label be raised appropriately.
- The following syntax should be accepted when the user can change both the information and associated sensitivity labels of an object:
  - New information label or changes - when changes are only to information label
  - [New sensitivity label or changes] - when changes are confined to sensitivity label
  - New information label or changes [New sensitivity label or changes] - when there are changes to both information and sensitivity labels, or when user wants to set sensitivity label to same level as information label.
- The user should be shown the label resulting from the changes and asked to confirm them before they are finally made by the system.
- The existence of classifications and words for which the user is not cleared should be hidden from the user by treating such classifications or words in a manner identical to classifications or words that are not defined in the encodings.
- Whenever a user enters multiple hierarchically related words in the same label, only the highest of the words should remain in the label.
- Whenever a user enters multiple words that cannot be combined in the same label, only the first of the words specified should remain in the label.
- To the maximum extent unambiguously possible, errors made in typing changes to labels should be corrected, with error messages to the user.

#### A.1.3.2 GUI

The following guidelines apply to GUIs that allow label changing via the selection of individual classifications and words. Whenever reference is made in this section to a mouse, other similar pointing devices (e.g., trackballs) are also acceptable.

- The graphical interface should be integrated with the typing interface, such that the user can specify changes using either the mouse or by typing.

- A character string representation of the label should be visible after each mouse selection.
- Only classifications and words that are valid for the user to select for a label or are required in the label should be displayed in association with that label.
- Each classification or word should be annotated to indicate whether or not the classification or word is present in the label.
- Each classification or word should be separately annotated if it cannot always be selected.
- Each change specifiable through typing should have an analogy in the GUI.
- When displayed as selections in the GUI, classifications should be visually separated from words, words should appear in the order specified in the encodings, and the first pure marking word should be visually separated from the previous words. These separations should be accomplished without identification of the various components on the display (e.g., classifications, compartments, markings) because there is no universally accepted identification terminology.

If all potential selections cannot fit on the screen, a scrolling or paging mechanism should be implemented to display all selections.

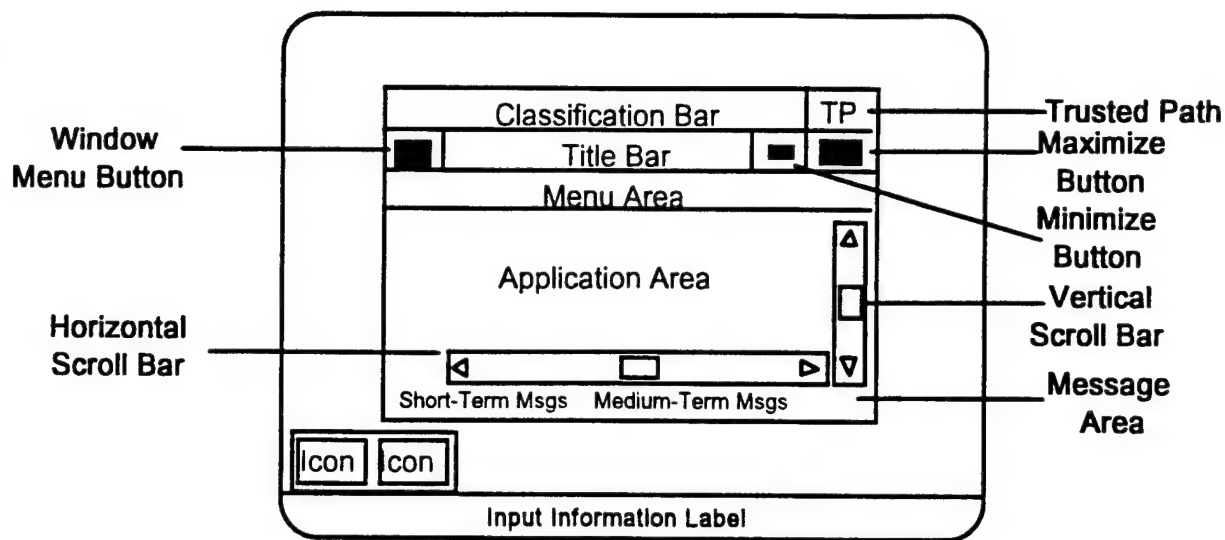
## **A.2 WINDOW STANDARDIZATION**

Window standardization is necessary to avoid confusion when users move from one CMW to another and to simplify training. To provide a standard user interface, CMW vendors will comply with the following guidance.

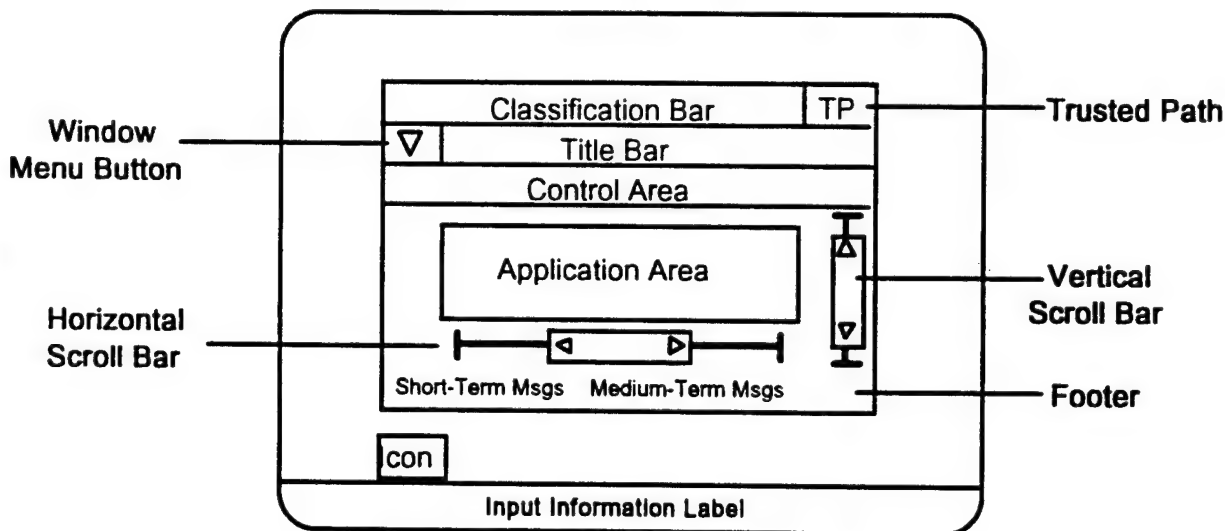
### **A.2.1 Screen Presentation**

- When the machine is turned on (and before CMW is booted), user will be presented with a distinctive screen (Trusted Path) either through color, screen marking, font, or combination thereof. The user authorized to boot the CMW will be presented with a prompt to enter an appropriate user-id and, if validated, password (see Figure A-1). After booting, a user log-in screen will be displayed.
- The CMW will be configurable so at least three windows can be displayed upon user authentication.
- Icons will default to the lower left portion of the screen, starting from the lower left corner and moving right as more icons are created.





### OpenLook



**Figure A-1. Sample CMW Screens**

#### A.2.2 Classification Bar

- A Classification Bar for output of sensitivity and information labels will be applied to each application base window created by the CMW. The Classification Bar will appear directly above the Title Bar.
- A Trusted Path button will be placed in the Classification Bar at the far right of the window. The Trusted Path button will be distinguishable by using the same mechanism (screen marking, color, font, or combination thereof) used for Trusted Path at initial log-in.

- If the information label, spacing separator, sensitivity label, and Trusted Path button are longer than the space provided in the Classification Bar, the window manager will provide the following default for trimming the labels:
  - Labels will be truncated from the right beginning with the information label and followed by the sensitivity label.
  - Truncation of the information label will be denoted by a dash followed by a greater than symbol “->” on the left side of the label. Truncation of the information label will stop when only one character of the label remains.
  - Truncation of the sensitivity label will be denoted by a less than symbol followed by a dash “<-” on the right side of the label. Truncation of sensitivity label will stop when only one character of the label remains, not including the left square bracket ( [ ).

Example:      TOP SECRET A B SA SB NOFORN [TS A B]  
                   ->TOP SECRET A B SA SB [TS A B]  
                   ->TOP SECRET [TS A B]  
                   ->TOP [TS A B]  
                   ->T [T<-

- The user must be able to view the entire information label by positioning the pointer in the classification bar and pressing any mouse button, or by menu selection via the Trusted Path.

### A.2.3 Reserved Area of the Screen

- One of the CMW Trusted Path requirements is that the CMW will “provide reserved portions of the screen to which user processes cannot write.” Normally, the Input Information Label will be displayed in the reserved area. Additionally, a visual indicator (e.g., blinking or highlighted Trusted Path symbol) should be displayed in this area when the Trusted Path is active (e.g., when a window classification is being changed). The Trusted path menu can be invoked by moving the pointer to the Reserved Area and depressing the Motif Custom or Open Look Menu mouse button.
- If the Input Information Label is too long for the space provided, the window manager will provide the following default for trimming the label:
  - Input Information Labels will be truncated from the right.
  - Truncation of the Input Information Label will be denoted by a dash followed by a greater than symbol “->” on the left side of the label.

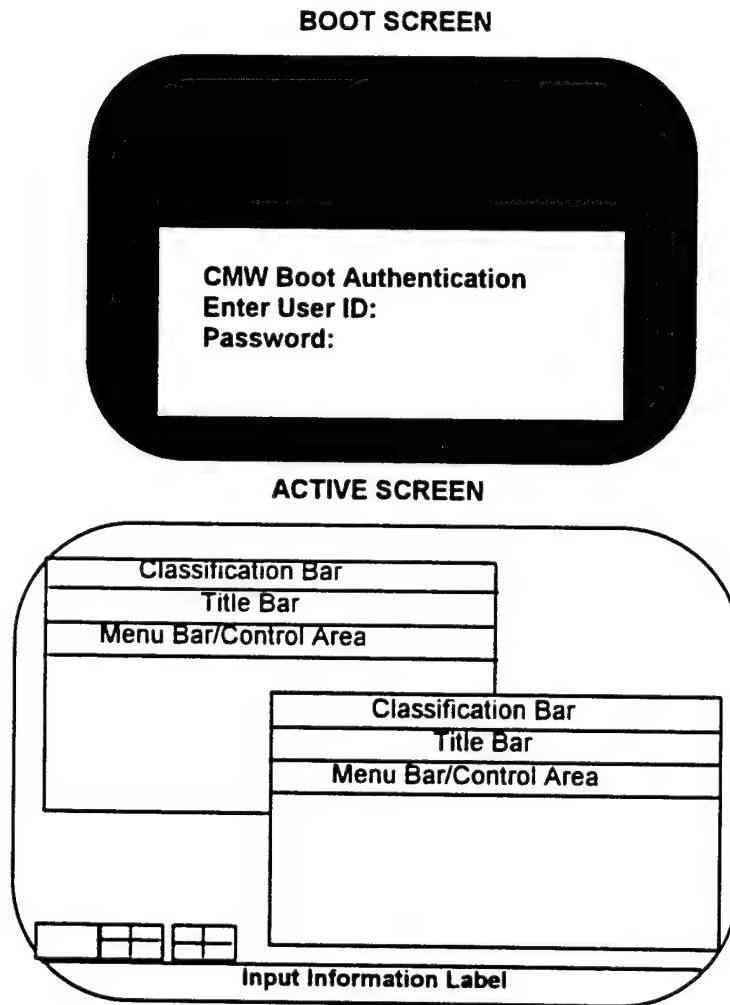
Example:      TOP SECRET A B SA SB  
                   ->TOP SECRET A B SA SB PROJECT X/Y LIMDIS ORCON O

- The user must be able to view the entire Input Information Label by positioning the pointer in the reserved area and pressing the select mouse button, or by menu selection via the

Trusted Path menu. Also, upon selecting the appropriate menu item, the user may change the Input Information Label for the active window using a dialog box.

#### A.2.4 Trusted Path Button

- When the user clicks on the Trusted Path button, a pop-up menu with the following minimum options will be displayed in the center of the screen (see Figure A-2):



**Figure A-2. Sample Trusted Path Main Menu**

- Create New Window (specifying the sensitivity level and input level or letting it default to the current window)
- Change Password
- Change Information Label of a file (not of the current window)

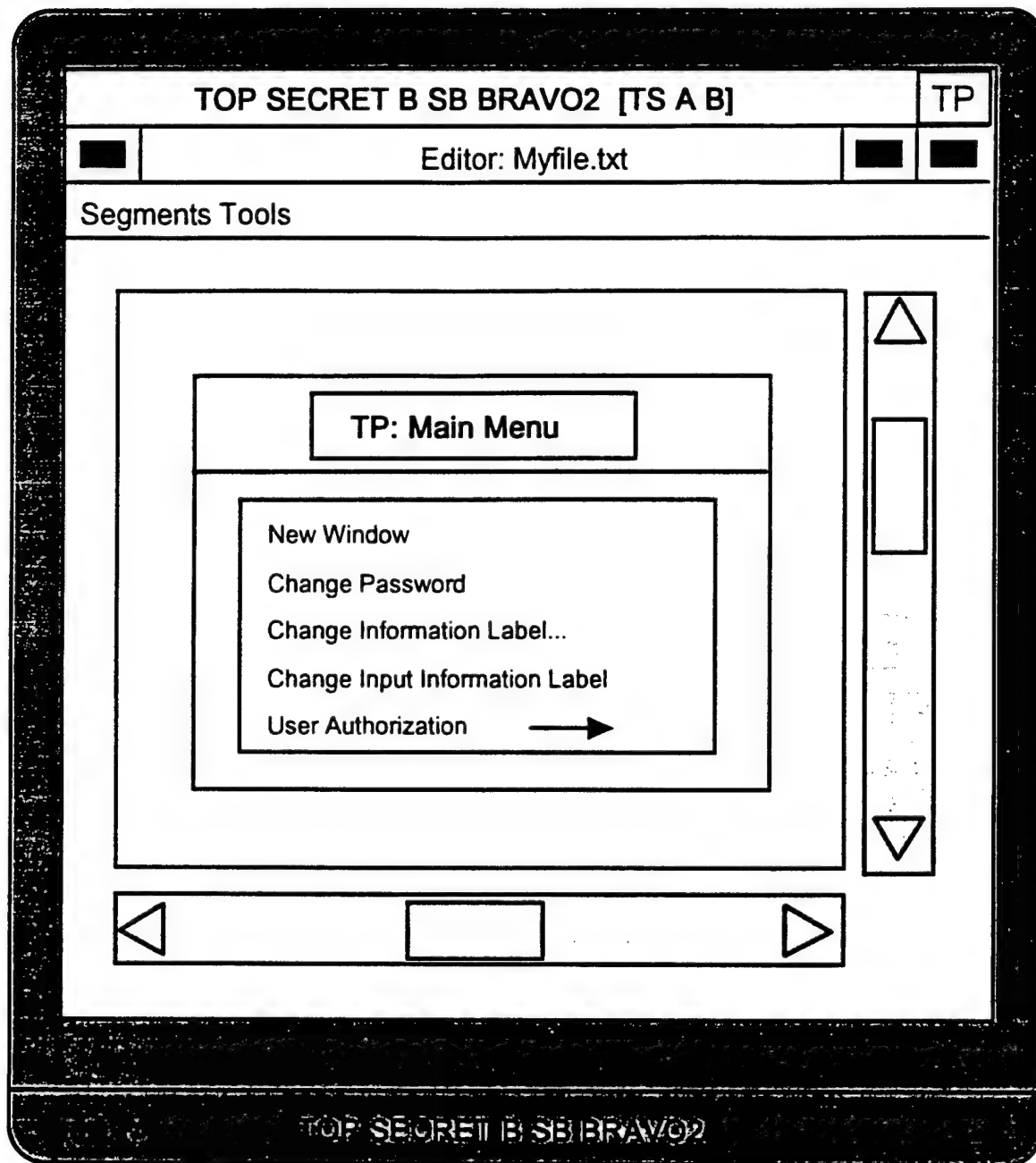
- Change Input Information Label
- User Authorization ->.
- When a user selects "User Authorization->," a cascading menu will be displayed. The contents of the menu will depend on the individual's USER-ID (whether a "normal" user, Information System Security Officer [ISSO], Systems Administrator, or Operator), and the authorizations given that individual from the Trusted Facility Management. If the user clicks outside the bounds of the user authorization menu, the display will return to the Trusted Path main menu.
- When an authorized normal user (one whom the ISSO has included in the Access Control List of the privileged program) selects "User Authorization->," a cascading menu with the following minimum authorizations will be displayed.
  - No Classification Marking - allows a user to bypass the requirement for printing information labels on the top and bottom of each page of printed output. (It does NOT permit the user to bypass the requirement for a print banner at the beginning and end of the output.)
  - Change Sensitivity Label - allows a user to change the sensitivity label of a file the user owned.
  - Set Sensitivity = Information Label - allows a user to set the sensitivity label of a file the user owns to the information label.
- When the ISSO selects "User "Authorization->," a cascading menu with the following minimum options will be displayed:
  - .. Change Sensitivity Label - allows the ISSO to set the sensitivity label of any file (file, device, file system, etc.) on the CMW.
  - Assign User Password/Clearance - allows the ISSO to set the password, password length, password time-limit, and clearance of a user's profile.
  - File Privileges - allows the ISSO to set/show the privileges associated with files.
  - Operator Administrator Authorization - allows the ISSO to set/show the authorizations for the other users, the administrator, and operators.
  - Set Audit Events - allows the ISSO to configure the auditing system.
  - Review Audit Data - allows the ISSO to reduce and review the audit data and store the audit data on removable media.
- When the System Administrator selects "User Authorization->," a cascading menu with the following minimum options will be displayed:

- Add User - allows the administrator to create a user profile, assign the user to a group, and create a user home directory. The ISSO must then assign the user a password.
- Add Group - allows the administrator to create a discretionary access group.
- ISSO Authorization - allows the administrator to set the authorizations of the ISSO.
- When the operator selects "User Authorization->," a cascading menu with the following minimum options will be displayed:
  - Make File System Backups - allows the operator to back up file system on the CMW.
  - Configure Printer - allows the operator to configure the printer (maximum sensitivity, printer definition, communication port, etc.).
  - Enable Printer - allows the operator to enable printer service to users.
  - Disable Printer - allows the operator to disable a printer for reconfiguration, etc.
  - Mount a file System - allows the operator to mount a file system on the CMW.
  - Unmount a File System - allows the operator to unmount a file system from the CMW.
  - Halt System - allows the operator to halt the CMW.

### **A.3 CLASSIFICATION DISPLAY ENHANCEMENTS**

This guide designates a minimum security functionality, which the vendor is encouraged to supplement. An example could be to provide a menu box from the Trusted Path that builds an information label for a specified file (see Figure A-3). Clicking a mouse button over the classification, multiple code words, multiple handling caveats, and multiple release markings could dynamically build the appropriate information label.

Another example could be to implement Trusted Path differently depending on whether it was invoked using the Trusted Path button or through the Trusted Path Background. The menu from the Trusted Path button could present options related to windows (Create, Cut/Paste, etc.) and the menu from the background selection could present non-window related items (Change Password, User Authorizations, etc.).



**Figure A-3. Sample for Entering New Input Information Label**

This page intentionally left blank.

**APPENDIX B**

**GLOSSARY**



This page intentionally left blank.

## APPENDIX B

### GLOSSARY

**AC:** See alternating current.

**accelerator key:** Special key or key combination that performs the same action as a menu selection

**Ada:** High-level computer programming language developed by the Department of Defense (DoD). Ada is used as the standard programming language for DoD. It is used for real-time processing, is modular in nature, and includes object-oriented features.

**alternating current (AC):** Electrical current that reverses its direction at regularly recurring intervals

**API:** Application Programming Interface

**APP:** Application Portability Profile

**application:** Classification of computer programs designed to perform specific tasks, such as word processing, database management, or graphics

**applications menu:** List of options within an application

**automated tools:** Software performing a sequence of operations to assist the user in achieving a goal (e.g., within graphics software, functions that align objects, smooth curves, or draw circles)

**backlighting:** Lit from the back. When referring to a monitor, light passes through the display screen from the back in order to illuminate screen images.

**base-level functions:** Initial or basic functions

**batch processing:** Processing data or the accomplishment of jobs accumulated in advance in such a manner that each accumulation thus formed is processed or accomplished in the same computer run

**baud:** Measure of the transmission speed capability of a communications line or system. In a sequence of binary signals, the rate of one baud equals one bit per second.

**bit-mapped display:** Display in which every picture element (pixel) of the screen can be referenced individually

**bookmarking:** Method of tagging items of interest to the user for easy referral later. Allows the user to customize the application.

**Boolean logic:** Logical expression that uses Boolean operators such as AND, OR, NOT, XOR, NOR, and NAND to create a statement that, when resolved, is either true or false

**Boolean operators:** A keyword in programming that causes two values to be combined in a logical fashion

**branching menu:** Menu that, if selected, brings up another menu

**bring-to-front:** Process of moving a window to the foreground

**Candela (cd):** Unit of luminous intensity expressed in Candela per square meter ( $\text{cd/m}^2$ ). One cd is equal to 0.29 footLambert.

**CAP:** Computer/Electronic Accommodation Program

**Cathode ray tube:** Electronic vacuum tube that focuses electrons energizing phosphors on a screen, creating a visible display. The typical computer monitor uses this type of display technology.

**cd:** See candela.

**central processor:** Portion of the computer that controls execution of applications

**character:** Single letter, digit, or symbol

**character string:** Series of alphanumeric characters, the contents of which are treated as though they were text

**CIE:** Commission International d'Eclairage

**CMW:** Compartmented Mode Workstation

**COBOL:** Acronym for Common Business-Oriented Language. COBOL is a computer programming language used extensively in mainframes and minicomputers for business applications.

**command:** Entry that instructs the computer to effect a specific action

**command entry:** Informing the computer that a specific command should be effected

**command icons:** Computer icons that represent frequently used computer commands and operations

**command language:** Limited programming language used strictly for executing a series of commands

**command stacking:** Allows the user to key a sequence of commands as a single “stacked” command entry

**compatible letters:** Letters easily associated with the function requested, for example, “P” for print, “Q” for quit

**context-sensitive:** Computer action or response directly related to the cursor position or specific point in the software, for example, a help function that displays information about the specific data entry field in which the cursor was located when help was called.

**control action:** Actions that must be effected to control a window or other graphics object or its contents

**control entry:** Input action by the computer user that changes some aspect of the appearance or function of the application

**control lockout:** Processing delay that results in pacing the capability to enter sequences of control commands

**COTS:** Commercial Off-The-Shelf (software)

**courseware:** Another name for educational or training materials and software

**CPU:** Central Processing Unit

**crosstalk:** Optical crosstalk, or bleeding, occurs when the light from the incorrect video image gets through. When referring to stereoscopic images, the right eye’s image is visible to the left eye or vice versa.

**CRT:** See Cathode Ray Tube

**CWS:** Compartmentalized workstation

**cursor:** Visual mechanism to mark, on-screen, where current input or output is to happen

**data entry:** Series of keystrokes used to input information into the computer

**data entry field:** Space (number of characters and/or digits) allowed for data entry

**data field:** Location in a file or database that contains a specific type of information

**database:** Structured or organized collection of information, which may be accessed by the computer

**database management system:** Computer application program that accesses or manipulates the database

**DBMS:** Database Management System

**DC:** See direct current.

**default:** Command that is automatically executed if none is specifically indicated

**default value:** Value of a variable in lieu of a specifically indicated value

**defeated:** Option that cannot be selected due to another selected option's use

**delimiter:** Symbols such as commas, spaces, or parentheses, which mark the boundaries of a specific block of information

**designate:** Process of selecting and displaying the current or active window with visual cues

**destructive entries:** Any entry that will destroy or overwrite information

**DIA:** Defense Intelligence Agency

**dialog:** Structured series of interchanges between a user and a computer terminal. Dialogs can be initiated by the computer or the user. Interactive dialog consists of an action by the user followed by a response from the computer or vice versa.

**dialog box:** Screen display box containing a message requesting additional information from the user

**direct current (DC):** Electrical current that flows in one direction only and is substantially constant in value

**direct manipulation:** Method of data organization (typically involving extensive windowing and iconization) in which the user can select specific displays of information and move them about to facilitate interaction with an application. A system of interaction in which the user's actions directly affect software operations.

**DISA:** Defense Information Systems Agency

**display frame:** Window or page

**display parallax:** When used in discussing touch screen technology, display parallax is the apparent displacement of an object viewed on a curved CRT screen and seen through a flat touch interactive display.

**display screen:** Screen of a multipage file

**DMA:** Defense Mapping Agency

**DoD:** U.S. Department of Defense

**DODIIS:** Department of Defense Intelligence Information Systems

**double keying:** Each character of the data item does not have an appropriately labeled key and therefore requires more than one keystroke for entry.

**DTED:** Digital Terrain and Elevation Data

**dual activation:** Two key are used simultaneously to input a command.

**dynamic depth displays:** Stereoscopic displays that are designed to change (move) images during viewing

**electroluminescence (EL):** Luminescence produced by electrical excitation of phosphor in powder or film form

**electronic mail:** Communication, processed through a network, from one workstation to another

**end user:** Person who ultimately uses the computer application or output

**error management:** Various options within an application that allow the user to eliminate the effects of commands executed accidentally or unwisely

**expand:** Ability to resize objects to produce better organization of on-screen material, usually a graphic or a window

**fc:** See footcandle.

**feedback:** Visual acknowledgment that the computer is executing the command or that the command was executed

**field:** Addressable data location

**file:** Any specifically identified collection of information stored in the computer

**FIP:** Federal Information Processing

**FIPS:** Federal Information Processing Standard

**FIRMR:** Federal Information Resources Management Regulation

**Fl:** See footLambert.

**footcandle (fc):** Unit of measurement of illumination. The amount of light emitted by a standard candle (1 cd) measured one foot away from the candle equals one footcandle.

**footLambert (Fl):** Unit of measure of intensity of reflected or emitted light (luminance). The average luminescence of any reflecting surface in footLamberts is the product of the illumination in footcandles by the luminous reflectance of the surface.

**frame:** Single display image or screen

**function key, fixed and variable:** Key which, when depressed, effects a specific action. It can either be a single, predefined function (fixed), or vary according to the system mode or level within an interactive dialog.

**form filling:** Method of interaction in which the user enters a series of commands or data items in predefined fields. These fields may be mandatory or optional.

**FORTRAN:** Acronym for FORMula TRANslator, which is a high level computer language used extensively in scientific and engineering applications

**freeze:** See Option - PAUSE

**GENSER:** general security

**GIS:** Geographic Information Systems

**GOTS:** Government Off-The-Shelf (software)

**graphical interaction:** Transactions between the user and computer-generated graphical representations of objects (screens, menus, buttons, etc.)

**Graphical User Interface (GUI):** System design that allows the user to effect commands, enter into transaction sequences, and receive displayed information through graphical representations of objects (menus, screens, buttons, etc.)

**GUI:** See Graphical User Interface.

**hard copy:** Printed copy of machine output in a visibly readable form, for example, printed reports, listings, documents, summaries

**hardware architecture:** Assemblage of a computer's internal components and its attached peripheral devices, which determine its capabilities and its limitations

**hatching:** Graphical pattern characterized by 45 and 135 degree diagonal lines that cross the patterned area

**HCI:** See Human Computer Interface.

**help screen:** Separate window that offers advice and information on how to overcome a specific problem and/or to better interact with the computer

**HFE:** See Human Factors Engineering.

**hierarchical menu:** Method of organizing menus in layers. The secondary or tertiary menus are stored within a primary menu.

**high level language:** Programming language that does not reflect the structure of any one computer or class of computers

**high resolution:** Screen display within an extremely fine visual reproduction of detail

**highlight:** Visual method to call attention to a specific piece of text or a graphic through differentiating it from surrounding texts or graphics. This is usually accomplished using contrasting colors or reverse video.

**hook:** Selecting a corner of a window or icon in order to move or resize it

**Human-Computer Interface (HCI):** Hardware and software allowing information exchange between the user and the computer

**Human Factors Engineering (HFE):** Approach that makes use of scientific facts in the design of items (i.e., computer systems, software, etc.) to produce effective human-machine integration and utilization

**icon:** Graphical representation of an object, concept, or message used by a computer system to represent items such as files, documents, programs, and disk drives.

**iconify:** Process that changes the text representation of an object, concept, or message into an icon

**iconification:** Process of iconifying



**IEEE:** Institute of Electrical and Electronics Engineers

**illuminance:** Measure of the quantity (density) of light reaching an object or surface. Measured in footcandles.

**Image Formation Time (IFT):** Measurement of the time required to update screen image displays

**Infrared (IR):** Radiation outside the visible light range on the red side (wavelength 0.75 to 0.8 micrometers)

**input focus:** Applies to a window that actually receives user input. This window is known as the active window where keyboard input appears and pointing device inputs apply. "Explicit" input focus refers to user or application action (e.g., typing keyboard accelerators, clicking pointer inside a window, moving a window through menu selection, etc.) to assign input focus. "Implicit" focus refers to focus automatically assigned to the window containing the location cursor.

**interactive control:** Attribute describing the ability of a program and a user to interface with each other during program execution

**interactive dialogue:** See dialog.

**interactive procedures:** Methods by which a user interacts with a computer and the computer with the user

**interface:** Interconnection and interrelationships between two devices, two applications, or the user and an application or device

**interlock:** Mechanism to connect two or more processes within a computing system to ensure that no one part of a hardware or software system can be operated independently

**interocular:** Perceptual coordination between the eyes

**IFT:** See image formation time.

**IR:** See infrared.

**JMCIS:** Joint Maritime Command Information System

**JPEG:** Joint Photographic Experts Group

**jump-ahead:** Capability of moving ahead during a step-wise process to allow quicker performance of an operation; useful for experienced computer users.

**justification:** Alignment of text on a display or a printed page. Left justification means that the left margin is even.

**keyword:** Special word in a programming language that tells the computer which operation to perform

**Lambert:** See footLambert.

**landscape:** Screen display or printing orientation parallel to the wide side of the paper

**LCD:** See Liquid Crystal Display.

**LCSS:** See Liquid Crystal Stereoscopic Shutter.

**left-justified:** See Justification.

**Liquid Crystal Display (LCD):** Display operated by polarizing light in which the nonactive segment reflects incident light and thus appears invisible against its background

**Liquid Crystal Stereoscopic Shutter (LCSS):** Type of display that utilizes liquid crystal shutters, one for each eye synchronized to alternate fields of the display, and representing one of the two images necessary to achieve the third dimension

**lockout:** Condition of the application locking the keyboard (i.e., not accepting commands from it) while the application is executing a command

**log on:** Process of gaining access to the system, usually involving a password and a recognition of the specific user by the computer

**logarithm:** The exponent that indicates the power to which a number has been raised to produce the given number:

$$N = 10^n \quad \log_{10} N = n$$

**luminance:** Amount of light per unit area reflected from or emitted by a surface. Measured in footcandles.

**lux:** Standard measure of illuminance. One lux is one lumen per square meter.

**macro:** Executable file that stores a series of commands and keystrokes to be used later

**MANpower and PeRsonnel INTeGration (MANPRINT):** An Army program that addresses concerns with manpower, personnel, training, human factors, system safety, and health hazards

**MANPRINT:** Acronym; MANpower and PeRsonnel INTEgration

**masking:** Partial or complete obscuring of one item by another

**memory:** Place in the computer in which information is stored

**menu:** List of options available within a software application

**menu bar:** The horizontal menu, usually at the top of the screen, which contains menu titles

**metaphor:** System-level analogy used for the grouping of processes and/or procedures. Usually associated with icons based on the analogy. As, for example, a desk top metaphor where icons represent office equipment or operations.

**minimize:** Procedure to make the window as small as it can be without being closed; this is usually done through iconization.

**mnemonic:** Word or code symbolic of another word, code, or function

**mode:** Status of the screen or program process

**Modulation Transfer Function (MTF):** A parameter using spatial frequency responses to characterize a screen display. The spatial frequency is stated in lines (line pairs) or minimum/maximum intensity pairs per unit distance. The MTF is used as a performance measurement of many optical systems.

**Motif:** User interface design approach based upon the “look” and “feel” presented in the OSF/Motif™ style guide. Motif™ is marketed by the Open Software Foundation.

**MTF:** See Modulation Transfer Function.

**multifunction keying:** Interface design where computer keys may perform multiple functions with the use of a combination of keystrokes

**multiwindow:** Simultaneous display of several windows on the computer screen

**natural language:** Programming language paradigm exemplified by using English-like commands and syntax to issue commands; interactions in the vernacular of the user.

**navigation:** Manner in which the user moves through the menu structure

**NATO Forces:** Personnel in the military forces of member nations of the North Atlantic Treaty Organization (NATO)

**NIST:** National Institute of Standards and Technology

**nit:** See normalized intensity.

**normalized intensity (nit):** Metric unit of measure of luminous intensity. A nit is equal to one candela per square meter ( $\text{cd/m}^2$ ) or 0.29 footLambert.

**null:** Empty; nothing. A null set contains no elements.

**OCR:** See Optical Character Recognition.

**one to many mapping:** An icon that represents a category of possibilities within an option is a one to many mapping.

**one to one mapping:** An icon that represents a single, specific function is a one to one mapping.

**OOP:** Object Oriented Programming

**Open Systems Environment:** See OSE.

**Open Software Foundation (OSF):** Consortium of computer hardware and software manufacturers whose membership includes over seventy of the computer industry's leading companies

**open window map:** A map (graphic display) that shows windows that are open and how they relate to each other

**open:** Procedure to cause a window to be displayed from an icon or menu option so that a document, directory or file can be viewed

**Optical Character Recognition (OCR):** The analysis and translation of a graphic representation of text into a coded form, such as ASCII or EBCDIC

**option:** Command that may be selected to access a specific function of an application

**option - BACKUP:** Option that will display the last transaction or the process of saving information to non-volatile memory

**option - CANCEL:** Command that allows the user to have the computer disregard a previous command

**option - CONFIRM:** Explicit warning of any possible data loss

**option - CONTINUE:** Option that resumes a transaction sequence which has been stopped by a PAUSE

**option - GOBACK:** Option that will display the last transaction. See also BACKUP.

**option - PAUSE:** Option that temporarily causes a transaction sequence to stop running. Use the CONTINUE option to resume after pausing.

**option - RESTART:** Option that will cancel any entries that have been made in a transaction sequence and return to the beginning of a sequence

**option - REVIEW:** Option that returns to the first display of a transaction sequence, allowing the user to review the transaction and make necessary changes

**option - SUSPEND:** Option that allows a user to leave the application, then, when he/she returns, resume at the same point he/she left off

**option - UNDO:** Option that immediately reverses any action

**option code:** Codes associated with the available choices

**OS:** Operating System

**OSE:** Open Systems Environment

**OSF:** See Open Software Foundation.

**output:** Information the computer displays in response to the user's actions

**overarching guidelines:** Dominant or all-embracing guidelines

**overlapping:** Windowing system in which one window covers a portion of another

**overlay:** Printing or drawing on a transparent or semi-transparent medium on the same scale as a map, chart, etc., to show details not appearing or requiring special emphasis on the original

**paging:** Scrolling through material one page at a time

**paired opposites:** Set of opposite functions, such as up and down, top and bottom

**pan:** Process to change the displayed region (often of a map) in a regular and smooth manner

**parallax:** Apparent displacement of an object as seen from two different points not on a straight line with the object

**parameter:** Quantity or constant whose value varies with the circumstances of the application

**piezoelectric:** Electric polarity due to pressure, especially in a crystalline substance

**pixel:** Contraction for picture element. A pixel is a single dot on a display screen.

**pixel matrix:** Arrangement of screen dots (pixels) to form text or graphic displays

**pop-up menu:** Lists of options that appear on the display screen in the form of a window

**portrait:** Screen or printing orientation parallel to the narrow side of the paper

**predictive modeling:** Use of a model to predict the actual response of a system or process

**preformatted:** Screen structure prepared for the user

**presentation graphics:** Pictorial representations of the relationships between variables (graphs and charts) or representations of systems (diagrams, schematics, and graphical renditions)

**primitive:** Code that defines a specific elementary shape, form, or color

**programming language:** Artificial language established for expressing computer programs

**prompt:** Text or graphic display that indicates the start point for user-generated actions. This term is also used for software generated instructions for process confirmation.

**pull-down menu:** Lists of options attached to a selection on a menu bar

**push-to-back:** Process of moving a window to the background

**QBE:** Query by Example

**QBF:** Query by Forms

**query language:** Specialized type of command language to elicit information from the computer system

**real time:** Absence of delay, except for the time required for transmission

**real-time control system:** Systems capable of responding to external events with negligible delays

**resize:** Procedure to change the size of a window or graphic

**resize border:** Window border that, if selected, allows user to resize the window

**resolution:** Density of picture display elements of the screen; degree of detail with which an image is displayed or printed.

**retrieve:** Procedure required to display stored information for purposes of viewing and manipulation

**RGB:** The original color display for the IBM PC (the Personal Computer Color Display IBM model 5151) used three discrete digital signals for each of the three primary colors. From these signals, the display type earned the nickname RGB from the list of additive primary colors: Red, Green, and Blue. Except for the interface signal, the RGB monitor works like a composite color monitor, using the same frequencies but substituting digital signals for analog.

**right-justified:** See justification.

**SAW:** See Surface Acoustic Wave.

**scroll:** Method used to move the contents of a window or list in a dialogue box using the scroll bar or scroll arrows

**scroll bar:** Rectangular bar that may be along the right edge or bottom of a window. Clicking or dragging in the scroll bar causes the view of the document to change.

**secondary coding:** Providing more than one method for coding displayed information. For example, in coding a particular item with red color, the use of the symbol "R" would provide secondary method for conveying the information when color was not available.

**semantics:** Relationship of characters or groups of characters to their meanings, independent of the manner of their interpretation and use

**sensitivity analysis:** Study that shows the response of a system to varying conditions. For example, "How sensitive is the system to increased workload?"

**sequence control:** Prescribed control over the order of function performed by the computer; this impacts the way in which a user interacts with the application.

**size coding:** Variations in the size of displayed alphanumerics and symbols. Such coding can be used for categorization.

**slider:** Part of the scroll bar that indicates what part of the file contained in a window is being viewed

**soft keys:** Visual representation of key functions on the display screen. This is usually associated with software controlled function key capabilities.

**specular reflector:** Reflecting light in a diffuse manner

**SQL:** Structured Query Language

**stacked command:** Single command composed of multiple commands that must be executed individually

**stereopsis:** Phenomenon of simultaneous vision with two eyes in which there is a vivid perception of distance of objects from the viewer (three-dimensional or stereoscopic vision)

**stereoscopic:** Method of seeing objects in three dimensions

**stroke width (sw):** Width of the line used to create a displayed character

**subordinate window:** A window that is opened from and controlled by another window

**subtend:** Opposite in position

**summary symbols:** Symbol that categorizes the information portrayed by a group of symbols

**supraordinate window:** Higher level window, usually the window from which subordinated options or tasks are controlled

**Surface Acoustic Wave (SAW):** When used in the context of touch screen technology, an approach that uses ultrasonic sound "beams" transmitted from two perpendicular sides of a display frame.

**sw:** See stroke width.

**system level menu:** List of which applications are available for utilization

**system response time:** Amount of time that elapses between a command being given and its being executed by the computer

**text editor:** Application that allows text to be created or modified

**text-based systems:** Method of organization in which the primary form of interaction between the system and user is through text rather than through graphical or voice interaction

**three-dimensional:** Relating to the three physical dimensions (height, width, depth). Giving the effect of depth or varying distances.

**TID:** See Touch Interactive Display.



**tiling:** Windowing approach in which multiple windows do not overlap, rather, all lie on the same plane.

**theater-type displays:** Display screens suitable for large group presentations as used in a movie theater or auditorium

**title banner:** Horizontal bar at the top of a window that shows the name of the window and allows it to be moved

**Touch Interactive Display (TID):** Uses a physical device between the user and the display which acts as the input mechanism

**transaction:** Interaction between a user and a computer in which the user inputs a command to receive a specific result from the computer

**transaction sequence:** Order of transactions required to accomplish the desired results

**transmissivity:** Measurement of the ability of an image to be transmitted. When used in the context of touch screen technology, refers to the ability of the image to be transmitted through a filter placed in front of a computer screen.

**type-ahead:** Capability of the computer to receive commands faster than it can display their results

**UAPI:** Uniform Application Program Interface

**UCI:** See User-Computer Interface

**UIDL:** User Interface Definition Language

**UIMS:** User Interface Management System

**user-callable:** Able to be requested by the user as desired

**User-Computer Interface (UCI):** Hardware and software allowing information exchange between the user and the computer

**user-specified windows:** Windows whose content has been selected by the user

**variable:** Quantity that can assume any of a given set of values

**VDT:** See Video Display Terminal.

**Video Display Terminal (VDT):** Terminal composed of a keyboard for data input and a CRT screen for display of the input/output

**widget:** Basic graphical object, which is a component of a user interface component

**window:** Typically rectangular display that provides a visual means for interaction with an application

**zoom:** Graphical tool used to magnify a portion of a document for more detailed viewing

This page intentionally left blank.

## APPENDIX C

### REFERENCES

*Note: References appearing in this section represent documents used in preparation of this volume, including some sources used at the time of initial document development that may no longer be current or applicable. The reader is advised to check the current applicability of a reference appearing in this list before using it as an information source. The reference section will be completely reviewed and revised for the next release of the TAFIM.*

41 CFR. *Federal Information Resources Management Regulations (FIRM)*, Chapter 201. Government Printing Office (GPO), Washington, D.C.

Abramson, S. R., L. H. Mason, and H. L. Snyder. 1983. "The Effects of Display Errors and Font Styles Upon Operator Performance with a Plasma Panel." In *Proceedings of the Human Factors Society - 27th Annual Meeting*, pp. 28-32. HFS, Santa Monica, California.

ACM - see Association for Computing Machinery, Inc.

Air Force Intelligence Agency. 1990. *Air Force Intelligence Data Handling System Style Guide*. Air Force Intelligence Agency, Washington, D.C.

Ambrosio, S., and K. Hooper. 1990. *Learning with Interactive Multimedia: Developing and Using Multimedia Tools in Education*. Microsoft Press, Redmond, Washington.

American Institute of Graphic Arts (AIGA). 1982. *Symbol Signs*. Visual Communications Books, Hastings House, New York.

Andrews, J. R., W. E. Haas, and M. D. Rainsdon. 1989. "Holographic Displays for the Man-Machine Interface." *Optical Engineering*, 28(6):643-649.

Andriole, S. J., and L. A. Adelman. 1990. "Prospects for Cognitive Systems Engineering." Presentation, Center for Multidisciplinary Information Systems Engineering, Drexel University, Philadelphia.

Antin, J. 1988. "An Empirical Comparison of Menu Selection, Command Entry and Combined Modes of Computer Control." *Behavior and Information Technology* 7(2):173-182.

Apple Computer, Inc. 1992. *Macintosh Human Interface Guidelines*. Addison-Wesley Publishing Company, Reading, Massachusetts.

Arend, U., K. Muthig, and J. Wandmacher. 1987. "Evidence for Global Feature Superiority in Menu Selection by Icons." *Behaviour and Information Technology* 6(4):411-426.

- Armstrong, H. G. 1988. "Colored Light-Emitting Diodes: Use of Red or Green in High Ambient Illumination." *In Engineering Data Compendium: Human Perception and Performance*, eds. K. R. Boff and J. E. Lincoln, Vol. 3:2260-2261. Harry G. Armstrong Aerospace Medical Research Laboratory, Wright-Patterson Air Force Base, Ohio.
- Arnstein, J. 1983. *The International Dictionary of Graphic Symbols*. Whitstable Litho Ltd., Whitstable, Kent, England.
- Aspillaga, M. 1991. "Screen Design: Location of Information and Its Effects on Learning." *Journal of Computer-Based Instruction* 18(3):89-92.
- Association for Computing Machinery, Inc. (ACM). 1990. *Resources in Human-Computer Interaction*. ACM Press, New York.
- Association for Computing Machinery, Inc. (ACM). 1989. *Human Factors in Computing Systems: CHI '89 Conference Proceedings*, Austin, Texas. ACM Press, New York.
- Auerbach Publishers, Inc., ed. 1981. *Practical Data Base Management*. Reston Publishing Company, Inc., Reston, Virginia.
- Avery, L. W., R. V. Badalamente, S. E. Bowser, P. A. O'Mara, and S. E. Reynolds. 1990. *Human Factors Design Guidelines for the Army Tactical Command and Control System (ATCCS) Soldier-Machine Interface, Version 1.0*. Pacific Northwest Laboratory (PNL) for the U.S. Army Tactical Command and Control System (ATCCS) Experimentation Site (AES), Fort Lewis, Washington.
- Avery, L. W., S. E. Bowser, S. M. Adams, R. V. Badalamente, D. T. Donohoo, D. A. Gellert, K. D. Hargrove, W. A. Hesser, J. G. Heubach, P. A. O'Mara, D. J. Pond, R. B. Randall, S. E. Reynolds, and M. S. Rowley. 1992. *Human Factors Design Guidelines for the Army Tactical Command and Control System (ATCCS) Soldier-Machine Interface, Version 2.0*, eds. L. W. Avery and S. E. Bowser. Pacific Northwest Laboratory (PNL) for the U.S. Army Tactical Command and Control System (ATCCS) Experimentation Site (AES), Fort Lewis, Washington.
- Aykin, N. M. and T. Aykin. 1991. "Individual Differences in Human-Computer Interaction." *Computers & Industrial Engineering* 20(3):373-379.
- Badler, N., and B. Webber. 1991. "Task Communication Through Natural Language and Graphics." *AI Magazine* 11(5):71-73.
- Baeker, R. 1980. "Towards an Effective Characterization of Graphical Interaction." *IFIP Workshop on Methodology of Interaction*, pp. 127-147.
- Baggen, E. A., H. L. Snyder, and M. R. Miller. 1988. "A Human Factors Evaluation of Current Touch-Entry Technologies." *In SID International Symposium Digest of Technical Papers*, pp. 259-262. SID, Playa Del Rey, Kingsbeach, California.

Bailey, M. J. 1993. "Guidelines For The Use Of Color In Scientific Visualization." SIGGRAPH Course Notes. San Diego Supercomputer Center.

Bailey, R. W. 1989. *Human Performance Engineering: Using Human Factors/Ergonomics to Achieve Computer System Usability*. Prentice Hall, Englewood Cliffs, New Jersey.

Bailey, R. W. 1982. *Human Performance Engineering: A Guide for System Designers*. Prentice Hall, Englewood Cliffs, New Jersey.

Baker, C. 1988. *Text-Editing Performance as a Function of Screen Size: A Pilot Study*. 612716.H7000700011, U.S. Army Human Engineering Laboratory, Aberdeen Proving Ground, Maryland.

Barnett, B. J. 1990. "Aiding Type and Format Compatibility for Decision Aid Interface Design." In *Proceedings of the Human Factors Society 34th Annual Meeting*, pp. 1552-1556. HFS, Santa Monica, California.

Barten, P. G. J. 1991. "Resolution of Liquid-Crystal Displays." In *Society for Information Display 91 Digest*, pp. 772-775. SID, New York.

Beaton, R. J. 1990. "Human Factors Engineering of 3D Stereoscopic Display Systems." In *Stereographics*, 17th International Conference on Computer Graphics and Interactive Techniques, Association for Computing Machinery, Special Interest Group on Graphics, Dallas, Texas.

Beaton, R. J., and S. T. Knox. 1987. "Flat Panel Image Quality." In *Society for Information Display 87 Digest*, pp. 115-118. SID, New York.

Benbasat, I., and Y. Wand. 1984. "A Structured Approach to Designing Human-Computer Dialogues." *International Journal of Man-Machine Studies* 21(2):105-126.

Benson, B. L., and J. E. Farrell. 1988. "The Effect of Character Height-to-Width Ratio on CRT Display Legibility." In *Society for Information Display 88 Digest*, pp. 340-343. SID, New York.

Biberman, L. M., and B. Tsou. 1991. *Image Display Technology and Problems with Emphasis on Airborne Systems*. Technical Report No. AD-B1157 161, Defense Technical Information Center, Alexandria, Virginia.

Bidgoli, H. 1989. "DSS Products Evaluation: An Integrated Framework." *Journal of Systems Management* 40(11):27-28

Billingsley, P. A. 1988. "Taking Panes: Issues in the Design of Windowing Systems." In *Handbook of Human-Computer Interaction*, ed. M. Helander, pp. 413-434. Elsevier Science Publishers B.V., Amsterdam.

Blaha, R. J. 1992. *Emerging Large Screen Display Technology*. Technical Report No. M92B0000010, The MITRE Corp., Bedford, Massachusetts.

Blaha, R. J. 1990. *Large-Screen Display Technology Assessment for Military Applications*. Technical Report No. M90-16, The MITRE Corp., Bedford, Massachusetts.

Blaha, R. J., C. D. Crotty, and D. F. Martin. 1992. *Evaluation of Large-Screen Display Systems in Workstation Development Environment*. Working Paper No. 92B0000081, The MITRE Corp., Bedford, Massachusetts.

Blankenberger, S., and K. Hahn. 1991. "Effects of Icon Design on Human-Computer Interaction." *Int. J. Man-Machine Studies* 35:363-377.

Blaser, A., and M. Zoeppriz, eds. 1983. *Enduser Systems and Their Human Factors*. Springer-Verlag, Berlin.

Blattner, M. M., and R. B. Dannenberg. 1992. *Multimedia Interface Design*, ACM Press, New York.

Blattner, M. M., D. A. Sumikawa, and R. M. Greenberg. 1989. "Earcons and Icons: Their Structure and Common Design Principles." *Human-Computer Interaction* 4:11-44.

Bobko, D. J., P. Bobko, and M. A. Davis. 1986. "Effect of Visual Display Scale on Duration Estimates." *Human Factors Society*, 28(2):153-158. HFS, Santa Monica, California.

Boehm-Davis, D. A., R. W. Holt, M. Koll, G. Yastrop, and R. Peters. 1989. "Effects of Different Data Base Formats on Information Retrieval." *Human Factors* 31(5):579-592.

Bosman, D. 1991. "Simulation of the Perception of a Computer-Generated Display Image." *Proceedings of Society for Information Display*, 32(1):3-12. SID, New York.

Bowser, S. E. 1991. "Review of Army Tactical Command and Control System Soldier-Machine Interface Functional Issues." Pacific Northwest Laboratory (PNL) Task Order 91-04, Subtask 2b for the U.S. Army Tactical Command and Control System (ATCCS) Experimentation Site (AES), Fort Lewis, Washington.

Breen, P. T. 1989. *Functional Requirements for C3I Large Screen Displays*. Technical Report No. M89-01, The MITRE Corp., Bedford, Massachusetts.

Breen, P. T. 1987. "Functional Requirements for C3I Large Screen Displays." In *Large Screen Projection Displays*, 760:2-8. SPIE Press, Bellingham, Washington.

Breen, P. T., and H. C. Masterman. 1990. *Modulation Depth Metrics for the Large Screen Displays in NASA's Mission Control Centers*. Working Paper No. 28736, The MITRE Corp., Bedford, Massachusetts.

Breen, P. T., P. E. Miller-Jacobs, and H. H. Miller-Jacobs. 1987. "Color Displays Applied to Command, Control, and Communications (C3) Systems." *In Color and the Computer*, ed. J. H. Durrett, pp.171-187. Academic Press, Inc., Boston.

Brockmann, R. J. 1990. *Writing Better Computer User Documentation - From Paper to Hypertext Version, 2.0*. John Wiley & Sons, Inc. New York.

Brooks, J. D., R. D. Gilson, and H. R. Myler. 1990. "Display Design Guide for Bivisual Media." *Proceedings of the Human Factors Society 34th Annual Meeting 1990*. HFS, Santa Monica, California.

Brosda, V., and G. Vossen. 1988. "Update and Retrieval in a Relational Database Through a Universal Schema." *ACM Transactions on Database Systems* 13(4):449-485.

Brosley, M., and B. Shneiderman. 1978. "Two Experimental Comparisons of Relational and Hierarchical Database Models." *International Journal of Man-Machine Studies* 10:625-637.

Brown, C. M. 1989. *Human-Computer Interface Design Guidelines*. Ablex Publishing Corporation, Norwood, New Jersey.

Brown, C. M., D. B. Brown, H. V. Burkleo, J. E. Mangelsdorf, R. A. Olsen, and R. D. Perkins. 1983. *Human Factors Engineering Standard for Information Processing Systems*. LMSC-D-877141 (Revision of C410), Lockheed Missiles and Space Company, Inc., Los Angeles.

Bullinger, H. J., K. P. Fährnich, and J. Ziegler. 1987. "Software-Ergonomics: History, State-of-the-Art, and Important Trends." *In Cognitive Engineering in the Design of Human-Computer Interaction and Expert Systems*, ed. G. Salvendy, pp. 307-316. Elsevier Science Publishers B.V., Amsterdam.

Bunnell, D., ed. 1993. *1993 Multimedia Tool Guide*. Newmedia Technologies for Desktop Computer Users, Special Issue: March 1993.

Burger, J. 1993. *The Desktop Multimedia Bible*. Addison-Wesley Publishing Company, Reading, Massachusetts.

Campbell, J. A., and S. P. Ross. 1987. "Issues in Computer-Assisted Interpretation of Graphs and Quantitative Information." *In Cognitive Engineering in the Design of Human-Computer Interaction and Expert Systems*, ed. G. Salvendy, pp. 473-480. Elsevier Science Publishers B.V., Amsterdam.

Carbone, R. M., and D. MacIver. 1987. "A Survey of Large-Screen Displays of C3I Applications." *In Large Screen Projection Displays*, 760:6-10. SPIE Press, Bellingham, Washington.

Carey, J. M., ed. 1988. *Human Factors in Management Information Systems*. Ablex Publishing Corporation, Norwood, New Jersey.



- Carlson, E. D., and W. Metz. 1980. "Integrating Dialog Management and Data Base Management." *Information Processing: Proceedings of the IFIP Congress*, Vol. 80, pp. 463-468. North-Holland Publishing Co., Amsterdam.
- Carroll, J. M., and J. McKendree. 1987. "Interface Design Issues for Advice-Giving Expert Systems." *Communications of the ACM* 30(1):14-31.
- Carroll, J. M., and S. A. Mazur. 1986. "LisaLearning." *Computer* 19(11):35-49.
- Chao, B. P. 1986. *Design Guidelines for Human-Computer Dialogues*. SAND86-0259, Sandia National Laboratories, Albuquerque, New Mexico.
- Chao, B. P. 1987. "Prototyping a Dialogue Interface: A Case Study." In *Cognitive Engineering in the Design of Human-Computer Interaction and Expert Systems*, ed. G. Salvendy, pp. 357-363. Elsevier Science Publishers B.V., Amsterdam.
- Chapnick, P. 1989. "Ushering in a New Era." *AI Expert*, Feb. 1989, p. 7-8, Miller Freeman, Inc.
- Chimera, R. 1993. "Evaluation of Platform Independent User Interface Builders." (HCIL 93-04) Human-Computer Interaction Laboratory, University of Maryland, College Park, Maryland.
- Christensen, C., E. A. Baggen, and H. L. Snyder. 1985. "Performance Measures and Subjective Evaluations for Two Color Displays." In *Proceedings of the Human Factors Society - 29th Annual Meeting*, pp. 1075-1078. Santa Monica, California.
- Clarke, A. A. 1986. "A Three-Level Human-Computer Interface Model." *International Journal of Man-Machine Studies* 24:503-517.
- Cowen, M. 1991. *A Comparison of Four Types of Feedback During Computer-Based Training (CBT)*. NPRDC-TR-92-2, Navy Personnel Research and Development Center, San Diego, California.
- Crolotte, A., J. Saleh, and A. Freedy. 1980. "Human Decision Processes in Command and Control of Marine Amphibious Brigade Operations," *IEEE*, pp. 1216-1220.
- Cuff, R. N. 1980. "On Casual Users." *International Journal of Man-Machine Studies* 12(2):163-187.
- Davies, S. P., A. J. Lambert, and J. M. Findlay. 1989. "The Effects of the Availability of Menu Information During Command Learning in a Word Processing Application." *Behavior and Information Technology* 8:135-144.
- Decker, J. J., C. J. Dye, C. J. C. Lloyd, and H. L. Snyder. 1991. *The Effects of Display Failures and Symbol Rotation on Visual Search and Recognition Performance*. Technical Memorandum 4-91. U.S. Army Human Engineering Laboratory, Aberdeen Proving Ground, Maryland.

Decker, J. J., P. L. Kelly, K. Kurokawa, and H. L. Snyder. 1991. *The Effect of Character Size, Modulation, Polarity, and Font on Reading and Search Performance in Matrix-Addressable Displays*. Technical Memorandum 6-91. U.S. Army Human Engineering Laboratory, Aberdeen Proving Ground, Maryland.

Defense Intelligence Agency (DIA). 1990. "DIA Standard Military Graphics Symbols Manual" (DIAM 65-xx) (Draft). DIA, Washington, D.C.

Defense Intelligence Agency (DIA). 1989. *Compartmented Mode Workstation Labeling: Source Code and User Interface Guidelines*. DIA Document DDS-2600-6215-89, DIA, Washington, D.C.

Defense Intelligence Agency (DIA). 1983. *DIA Standard User Interface Style Guide for Compartmented Mode Workstations*. DIA Memorandum U-15, 284/DSE-3, Washington, D.C.

Defense Information Systems Agency/Center for Information Management (DISA/CIM). 1994. *Technical Reference Model for Corporate Information Management, Version 2.0*. DISA/CIM, Washington, D.C.

Defense Information Systems Agency/Center for Information Management (DISA/CIM). 1993. *Department of Defense Technical Architecture Framework for Information Management, Volume 8. Human Computer Interface Style Guide, Version 3.0*. DISA/CIM, Washington, D.C.

Defense Information Systems Agency/Center for Information Management (DISA/CIM). 1992a. *Comparison of the IEEE Recommended Practice for Graphical User Interface Drivability with the DISA Center for Information Management Human Computer Interface Style Guide*. Technical Memorandum, DISA/CIM, Washington, D.C.

Defense Information Systems Agency/Center for Information Management (DISA/CIM). 1992b. *Human Computer Interface Style Guide, Version 1.0*. DISA/CIM, Washington, D.C.

Defense Information Systems Agency/Center for Information Management (DISA/CIM). 1992c. *Human Computer Interface Style Guide, Version 2.0*. DISA/CIM, Washington, D.C.

Diethelm, W., and M. Diethelm. 1984. *Signet Signal Symbol, Handbook of International Signs*. ABC Edition, Zurich.

Doll, T. J. 1991. "Electro-Optic/Infrared Technology and the Human-Machine Interface." In *Proceedings of the Human Factors Society - 35th Annual Meeting*, pp. 1500-1501. HFS, Santa Monica, California.

Dominessy, M. E. 1989. *A Literature Review and Assessment of Touch Interactive Devices*. Technical Memorandum 11-89. U.S. Army Human Engineering Laboratory, Aberdeen Proving Ground, Maryland.

Dreyfuss, H. 1984. *Sourcebook to International Graphic Symbols*. Van Nostrand Reinhold Company, Inc., New York.

Dumas, J. S. 1988. *Designing User Interfaces for Software*. Prentice Hall, Englewood Cliffs, New Jersey.

Durrett, H. J., ed. 1987. *Color and the Computer*. Academic Press, Inc., Boston.

Dye, C. J., and H. L. Snyder. 1991. *The Effects of Display Failures, Polarity, and Clutter on Visual Search for Symbols on Cartographic Images*. Technical Memorandum 9-91, U.S. Army Human Engineering Laboratory, Aberdeen Proving Ground, Maryland.

Eberts, R. E. 1994. *User Interface Design*. Prentice Hall, Englewood Cliffs, New Jersey.

Egan, D. E. 1988. "Individual Differences in Human-Computer Interaction." In *Handbook of Human-Computer Interaction*, ed. M. Helander, pp. 543-568. North Holland: Elsevier Science Publishers, B.V., Amsterdam.

Ehrenreich, S. L. 1981. "Query Languages: Design Recommendations Derived from the Human Factors Literature." *Human Factors* 23(6):709-725.

Ehrhart, L. S. 1990. "New Approaches to Human-Computer Interaction Research and Design for Decision Aiding Systems." In *Proceedings of the 5th IEEE International Symposium on Intelligent Control*. IEEE Computer Society Press, Los Alamitas, California.

Eisen, H. A. 1990. "Iconer: A Tool for Evaluating Icons." *SIGCHI Bulletin* 21(3):23-25.

Elkerton, J. and R. C. Williges. 1989. "Dialogue Design for Intelligent Interfaces." *Intelligent Interfaces: Theory, Research, and Design*, eds. P. A. Hancock and M. H. Chignell.

Elkerton, J., R. C. Williges, J. A. Pittman, and J. Roach. 1982. "Strategies of Interactive File Search" in *Proceedings of the Human Factors Society*. HFS, Santa Monica, California.

Farooq, M. U., and W. D. Dominick. 1988. "A Survey of Formal Tools and Models for Developing User Interfaces." *International Journal of Man-Machine Studies* 29:479-496.

Feldman, M. B., and G. T. Rogers. 1982. "Toward the Design and Development of Style-Independent Interactive Systems." In *Human Factors in Computer Systems: Proceedings*, March 15-17, 1982, pp. 111-116. ACM, New York.

Fenchel, R. 1981. "An Integrated Approach to User Assistance." *ACM SIGSOC Bulletin* 13 2:98-104.

Fernandes, K. 1992. *User Interface Specifications for Navy Command and Control Systems, Version 1.1*. U.S. Department of the Navy; Naval Command, Control, and Ocean Surveillance Center, Research, Development, Test, and Evaluation Division, San Diego, California.

Fernandes, K., and K. E. Maracle. 1991. *Design Standards for the Command and Control Information Navigation and Training System (CINTS)*. Technical Note 1660, Naval Ocean Systems Center, San Diego, California.

Finke, D. J., and M. M. Lloyd. 1988. "Guidelines and Tools for Human-Decision Aid Interface Design and Evaluation." In *Proceedings of the Fifth Annual Workshop on Command and Control Decision Aiding*, pp. 170-193, September 1988.

FIPS - See National Institute of Standards and Technology (NIST)

Flynn, M. K. 1993. "Multimedia PC Gets Updated." *PC Magazine*. 12(13):30.

Flynn, R. R. 1987. *An Introduction to Information Science*. Marcel Dekker, Inc., New York.

Fowler, S. L., and V. R. Stanwick. 1995. *The GUI Style Guide*. AP Professional, Imprint of Academic Press, Inc., Cambridge, Massachusetts.

Fowler, C. J. H., L. A. Macaulay, and J. F. Fowler. 1985. "The Relationship Between Cognitive Style and Dialogue Style: An Explorative Study." In *People and Computers: Designing the Interface*, eds. P. Johnson and S. Cook, pp. 186-198. Cambridge University Press, Cambridge.

Frost, R. A. 1984. *Database Management Systems*. McGraw-Hill, New York.

Frutiger, A. 1980. *Type Sign Symbol*. ABC Verlag, Zurich.

Funk, H. L. 1989. "Information Display - An Overview and Trends." In *Proceedings of the Institution of Electrical and Electronics Engineers*, pp. 2-1 - 2-7. IEEE, Washington, D.C.

Ga Cote, R. 1992. "Code on the Move." *Byte Magazine*. Jul 92, pp.206-226.

Gaines, B. R., and M. L. G. Shaw. 1986. "Foundation of Dialog Engineering: The Development of Human-Computer Interaction." Part II. *International Journal of Man-Machine Studies* 24:101-123.

Gaines, B. R., and P. V. Facey. 1975. "Some Experience in Interactive System Development and Application." In *Proceedings of the IEEE* 63(6):894-911.

Galitz, W. O. 1994. *It's Time To Clean Your Windows*. John Wiley & Sons, Inc., New York.

Galitz, W. O. 1993. *User-Interface Screen Design*. John Wiley & Sons, Inc., New York.

Galitz, W. O. 1989. *Handbook of Screen Format Design*. QED Information Sciences, Inc., Wellesley, Massachusetts.

Galitz, W. O. 1984. *Humanizing Office Automation*. QED Information Sciences, Inc., Wellesley, Massachusetts.

Garner, K. H. 1990. "20 Rules for Arranging Text on a Screen." *CBT Directions*:13-16.

- General Services Administration (GSA). 1991. *Managing Information Resources for Accessibility*. GSA, Clearinghouse on Computer Accommodation (COCA) of the Information Resources Management Service (IRMS), Washington D.C.
- Gery, G. 1991. *Electronic Performance Support Systems*. Weingarten Publications, Boston, Massachusetts.
- Getler, R. 1991. "The Case for Concurrent Authoring." *CBT Directions*:14-22.
- Gilmore, W. E., D. I. Gertman, and H. S. Blackman. 1989. "The User-Computer Interface in Process Control." In *A Human Factors Engineering Handbook*, pp. 21-33. EG&G Idaho, Inc., Idaho Falls, Idaho.
- Gittins, D. 1986. "Icon-Based Human-Computer Interaction." *International Journal of Man-Machine Studies* 24:519-543.
- Glasser, J., and A. Rolland. 1989. "Visual Performance Evaluation for LCD Displays: Appropriate Methods for Measuring Luminance and Contrast." *Proceedings of Society for Information Display*, 1077:9-20. SID, New York.
- Gold, R. S., and A. G. Ledebuhr. 1985. "Full Color Liquid Crystal Light Valve Projector." In *Advanced in Display Technology I*, 526:51-58. SPIE Press, Bellingham, Washington.
- Goldberg, A., and D. Robson. 1985. *Smalltalk-80: The Language and its Implementation*. Addison-Wesley, Reading, Massachusetts.
- Goode, W. F. 1991. "Status of Electronic Displays." Seminar for the Society of Information Display, SID, New York.
- Gordon, S. E. 1988. "The Human Factor in Expert Systems," *AI Expert*, pp. 55-59.
- Grabinger, R. S., and D. Amedeo. 1988. "CRT Text Layout: Perceptions of Viewers." *Computers in Human Behavior* 4:189-205.
- Green, P., and W. T. Burgess. 1980. *Debugging a Symbol Set for Identifying Displays: Production and Screening Studies*. Highway Safety Research Institute, University of Michigan, Ann Arbor, Michigan.
- Greenberg, S., and I. H. Witten. 1985. "Adaptive Personalized Interfaces -- A Question of Viability." *Behaviour and Information Technology* 4(1):31-45.
- Greitzer, F. 1987. Comments on draft NASA Guidelines Document: *User/Computer Interaction Section*. Memo on 1, 22, 1987.
- Grether, W. F., and C. A. Baker. 1972. *Human Engineering Guide to Equipment Design*, eds., H. P. Van Cott and R. C. Kinkade. GPO, Washington, DC.

Grill, E. 1990. *Relational Databases: A Methodical Guide for Practical Design and Implementation*. Ellis Harwood, New York.

GSA - See General Services Administration

Gunderson, J., G. Gruetzmacher, and N. Swanson. 1991. "Legibility of Seven Segment Numeric LED Displays: Comparisons of Two Fonts at Various Distances." In *Proceedings of the Human Factors Society - 35th Annual Meeting*, pp. 491-495. HFS, Santa Monica, California.

Hagan, T. 1992. "GUI Tools Boost Portability." *Open Systems Today* 28 Sep 92, pp.39-44.

Hamel, C. J., and S. L. Clark. 1986. *CAI Evaluation for the Design of Computer-Aided Instruction*. Technical Report NTSC TR86-002 (DTIC No. AD-A172383). Naval Training Systems Center, Orlando, Florida.

Hannigan, S., and V. Herring. 1987. "Human Factors in Office Product Design -European Practice." In *Proceedings of the Second International Conference on Human-Computer Interaction*, Conference Title "Cognitive Engineering in the Design of Human-Computer Interaction and Expert Systems," ed. G. Salvendy, pp. 226-232. Elsevier Science Publishers B.V., Amsterdam.

Hansen, G. W., and J. V. Hansen. 1992. *Database Management and Design*. Prentice Hall, Englewood Cliffs, New Jersey.

Hansen, W. J. 1971. "User Engineering Principles for Interactive Systems." *Proceedings Fall Joint Computer Conference*, pp. 523-543. AFIPS Press, Montvale, New Jersey.

Harper, E. 1992. "WinLogin Offers Control Over Configuration Files." *LAN Times* 9(21):92.

Harrell, T. H. 1987. "Designing User-Computer Dialogues: Basic Principles and Guidelines." In *Computerized Psychological Testing: Current Issues and Future Directions*, Honaker, L. M., Chair. Symposium conducted at the American Psychological Association, New York.

Harter, S. P. 1986. *On-line Information Retrieval*. Academic Press, Orlando, Florida.

Hawkrigde, D. G. 1988. *Computers in Company Training*. Croom Helm, London.

Heines, J. M. 1984. *Screen Design Strategies for Computer-Assisted Instruction*. Digital Press, Bedford, Massachusetts.

Helander, M., ed. 1988. *Handbook of Human-Computer Interaction*. Elsevier Science Publishers B.V., Amsterdam.

Henry, T. R., and S. E. Hudson. 1990. "Multidimensional Icons." *ACM Transactions on Graphics* 9(1):133-137.

Hershman, R. L., R. T. Kelly, and H. G. Miller. 1979. *User Performance With a Natural Language Query System for Command Control*. Navy Personnel Research and Development Center, San Diego, California.

HFS - See Human Factors Society

Hildreth, C. R. 1982. *Online Public Access Catalogs: The User Interface*. Online Computer Library Center, Inc. Dublin, Ohio.

Hix, D. 1991. "An Evaluation Procedure For User Interface Development Tools, Version 2.0." Virginia Polytechnic Institute and State University, Blacksburg, Virginia.

Hix, D., and R. S. Schulman. 1991. "Human-Computer Interface Development Tools: A Methodology For Their Evaluation." *Communications of the ACM* 34(3):75-87.

Hoadley, E. D. 1990. "Investigating the Effects of Color." *Communications of the ACM* 33(2):120-127.

Holtzman, S. 1989. *Intelligent Decision Systems*. Addison-Wesley, Reading, Massachusetts.

Horton, W. 1994. *The Icon Book, Visual Symbols for Computer Systems and Documentation*. John Wiley & Sons, New York.

Horton, W. K. 1990. *Designing and Writing Online Documentation - Help Files to Hypertext*. John Wiley & Sons, Inc. New York.

Human Factors Society, Inc. (HFS). 1988. *American National Standard for Human Factors Engineering of Visual Display Terminal Workstations*. ANSI/HFS 100-1988, HFS, Santa Monica, California.

Humphrey, S. M., and B. J. Melloni. 1986. *Databases: A Primer for Retrieving Information by Computer*. Prentice Hall, Englewood Cliffs, New Jersey.

Hunter, M. W. 1988. "CRT Anti-glare Treatments, Image Quality, and Human Performance." Ph.D. Dissertation, Virginia Polytechnic Institute and State University, Blacksburg, Virginia.

Hurd, J. C. 1983. "Writing Online Help." In *Proceedings of 30th International Technical Communication Conference*, pp. 151-154. Society for Technical Communication, Washington, D.C.

Hurley, A. F., and L. I. Hoffberg. 1991. *Evaluation of Six Large-Screen Displays*. Technical Report No. 29399, The MITRE Corp., Bedford, Massachusetts.

Hussein, B. 1989. "DSS Products Evaluation: An Integrated Framework." *Journal of Systems Management*, pp. 27-34, November 1989.



Hyland, R. M., and P. R. Boivin. 1987. *Performance Characteristics of Infrared Touch Screens for Typical Workstation Applications*. Technical Report No. NSU 8093, Defense Technical Information Center, Alexandria, Virginia.

IBM. 1984. *Human Factors Workstations with Visual Displays*, Third Edition. San Jose, California.

Inmon, W. H. 1981. *Effective Data Base Design*. Prentice Hall, Englewood Cliffs, New Jersey.

Innocent, P. R. 1982. "Towards Self-Adaptive Interface Systems." *International Journal of Man-Machine Studies* 16:287-299.

Institute of Electrical and Electronics Engineers (IEEE). 1993a. "Draft Standard for Information Technology-Uniform Application Program Interface-Graphical User Interfaces, "IEEE P1201.1, Draft 7. Institute of Electrical and Electronics Engineers Standards Department, Piscataway, New Jersey.

Institute of Electrical and Electronics Engineers (IEEE). 1993b. "IEEE Recommended Practice for Graphical User Interface Drivability," IEEE P1201.2, Draft 2. Institute of Electrical and Electronics Engineers Standards Department, Piscataway, New Jersey.

Institute of Electrical and Electronics Engineers (IEEE). 1993c. *Standards for Information Technology -- X Window System -- Modular Toolkit Environment (MTE)*, IEEE P1295. Institute of Electrical and Electronics Engineers Standards Department, Piscataway, New Jersey.

Institute of Electrical and Electronics Engineers (IEEE). 1992. "Draft Standard for Information Technology-Uniform Application Program Interface-Graphical User Interfaces," IEEE P1201.1, Draft 3.1. Institute of Electrical and Electronics Engineers Standards Department, Piscataway, New Jersey.

International Organization for Standardization (ISO). 1991. "Part 3 Flat Panel Addendum, Additional Notes for Reviewers." In "ISO 9241, Working Draft of ISO Committee 159 SC 4 Working Group 2." ISO 9241, ISO, Washington, D.C.

Jarvenpaa, S. L., and G. W. Dickson. 1988. "Graphics and Managerial Decision-Making: Research Based Guidelines." *Communications of the ACM* 31(6):764-775.

Jorna, R. 1988. "Chapter 10. A Comparison of Presentation and Representation: Linguistic and Pictorial." In *Human-Computer Interaction: Psychonomic Aspects*, eds. G. C. van der Veer and G. Mulder, pp. 172-185. Springer-Verlag, Berlin Heidelberg.

Karon, P. 1992. "Cross-Platform Tools Appeal to Developers." *InfoWorld* 17 Aug 92, pp. S74-S75.

Katzeff, C. 1986. "Dealing With a Database Query Language in a New Situation." *International Journal of Man-Machine Studies* 5(1):1-17.



Katzeff, C. 1988. "The Effect of Different Conceptual Models Upon Reasoning in a Database Query Writing Task." *International Journal of Man-Machine Studies* 29(1):37-62.

Kawamura, T., H. Kawamura, K. Kobara, A. Saitoh, and Y. Endo. 1991. "Anti-reflection Coating for Inner Surface of CRT Faceplate." In *Society for Information Display 91 Digest*, pp. 49-52. SID, New York.

Kearsley, G. 1988. *Online Help Systems: Design and Implementation*. Ablex Publishing, Norwood, New Jersey.

Kelley, J. F. 1984. "An Iterative Design Methodology for User-Friendly Natural Language Office Information Applications." *ACM Transactions on Office Information Systems* 2(1):26-41.

Kelster, R. S., and G. R. Galloway. 1983. "Making Software User Friendly: An Assessment of Data Entry Performance." *Proceedings of the Human Factors Society*. HFS, Santa Monica, California.

Kieras, D. E., and S. Boviar. 1984. "The Role of a Mental Model in Learning to Operate a Device." *Cognitive Science* 8:255-273.

Klien, G. A., and D. MacGregor. 1988. "Knowledge Elicitation of Recognition-Primed Decision-Making." *Technical Report 799*. U.S. Army Research Institute Field Office, Fort Leavenworth, Kansas.

Kobara, S. 1991. *Visual Design with OSF/Motif*. Addison-Wesley Publishing Company, Reading, Massachusetts.

Korth, H. F., A. Silbershatz. 1986. *Database System Concepts*. McGraw-Hill, New York.

Krause, J. 1980. "Natural Language Access to Information Systems: An Evaluation Study of Its Acceptance by End Users." *Information Systems* 5(4):297-318.

Kubota, H., T. Marushige, C. M. Gomes, and S. Kobayashi. 1988. "Legibility of Multiplexed Dot-Matrix LCDS: The Effect of Surface Reflection." In *Proceedings of Society for Information Display (SID)*, 29(3):213-216. SID, New York.

Kubota, H., T. Marushige, T. Takabayashi, and S. Kobayashi. 1986. "LCD Legibility." In *Society for Information Display 88 Digest*, pp. 157-160. SID, New York.

Kuwayama, Y. 1973a. *Trademarks & Symbols, Volume 1: Alphabetical Designs*. Van Nostrand Reinhold Company, New York.

Kuwayama, Y. 1973b. *Trademarks & Symbols, Volume 2: Symbolical Designs*. Van Nostrand Reinhold Company, New York.

Langen, M., B. Thull, T. Schecke, G. Rau, and G. Kalff. 1989. "Prototyping Methods and Tools for the Human-Computer Interface Design of a Knowledge-Based System." *Designing*

*and Using Human-Computer Interfaces and Knowledge-Based System: Proceedings of the Third International Conference on Human-Computer Interaction*, Vol. II, eds. G. Salvendy and M. J. Smith, pp. 861-868. September 18-22, 1989, Boston. Elsevier Science Publishers B.V., Amsterdam.

Lansdale, M. W. 1988. "On the Memorability of Icons in an Information Retrieval Task." *Behaviour and Information Technology* 7(2):131-151.

Lanzetta, T. M., and N. D. Lubart. 1988. "Comparing the Readability of Display Technologies: Paper, CRT, and LCD." In *Society for Information Display 88 Digest*, pp. 336-339. SID, New York.

Larson, H. T. 1989. "Large Automated Flat Situation Display for the Commander." *Maintaining Start-of-the Art in ACCS*:3-44. Army Science Board Summer Study.

Laurel, B., ed. 1990. *The Art of Human Computer Interface Design*. Addison-Wesley, Reading, Massachusetts.

Laverson, A., K. Norman, and B. Shneiderman. 1987. "An Evaluation of Jump-Ahead Techniques in Menu Selection." *Behavior and Information Technology* 6(2):97-108.

Laycock, J. 1985. "The Legibility of Passive Displays". *Proceedings of Society for Information Display*, 26(2):89-93. SID, New York.

LeMay, M. 1988. "Why Some Decision Aids Work and Others Do Not." *Proceedings of the 1988 IEEE International Conference on Systems, Man, and Cybernetics*, pp. 227-229.

Lerner, N. D., and B. L. Collins. 1980. *The Assessment of Safety Symbol Understandability by Different Testing Methods*. U.S. Department of Commerce, Washington, D. C.

Lewis, H. V., and J. J. Fallesen. 1989. *Human Factors Guidelines for Command and Control Systems: Battlefield and Decision Graphics Guidelines*. Research Project 89-01. U.S. Army Research Institute for the Behavioral and Social Sciences, Alexandria, Virginia.

Lickteig, C. W. 1989. *Design Guidelines and Functional Specifications for Simulation of the Battlefield Management Systems (BMS) User Interface*. U.S. Army Research Institute for the Behavioral and Social Sciences, Alexandria, Virginia.

Lloyd, C. J. C., J. J. Decker, and H. L. Snyder. 1991. *The Effects of Line and Cell Failures on Reading and Search Performance Using Matrix-Addressable Displays*. Technical Memorandum 7-91. U.S. Army Human Engineering Laboratory, Aberdeen Proving Ground, Maryland.

Lloyd, C. J. C., J. J. Decker, K. Kurokawa, and H. L. Snyder. 1988. "Effects of Line and Cell Failures on Reading and Search Performance Using Matrix-Addressable Displays." In *Society for Information Display 88 Digest*, pp.344-346. SID, New York.

Lochovsky, F. H, and D. C. Tsichritzis. 1984. "Querying External Databases." In *Human Factors and Interactive Computer Systems*, ed. Y. Vassiliou. Ablex Publishing Corporation, Norwood, New Jersey.

Lodding, K. N. 1983. "Iconic Interfacing." *IEEE Computer Graphics and Applications* 3(2):11-20.

Luther, A. C. 1992. *Designing Interactive Multimedia*. Bantam Books, New York.

Lysaght, R. J., R. Harris, and W. Kelly. 1988. "Artificial Intelligence for Command and Control." *Technical Report 2122*. U.S. Army Communications and Electronics Command, Fort Monmouth, New Jersey.

Ma, P., F. H. Murphy, and E. A. Stohr. 1989. "A Graphics Interface for Linear Programming," *Communications of the ACM* 32:996-1012.

MacGregor, J. M., and E.S. Lee. 1988. "A Feature Matching Approach to the Retrieval of Graphical Information." *Behavior and Information Technology* 7(4):457-465.

MacGregor, R. C. , K. F. King, and R. J. Clarke. 1988. "Individualising the Man-Machine Interface." In *Ergonomics of Hybrid Automated Systems, Proceedings of the First International Conference on Ergonomics of Advanced Manufacturing and Hybrid Automated Systems*, eds. W. Karwowski, H. R. Parsaei, and M. R. Wilhelm, pp. 275-281. Louisville, Kentucky.

Magnavox, Inc. 1985. *Appendix B - Application of Data Processing Networking Techniques for Army Command and Control Systems (ACCS) Task 1 Subtask 4*. Report No. 11-89 on Contract No. DAAK11-84-D-0006, Magnavox, Inc., Fort Wayne, Indiana.

Main, R. and D. Paulson. 1988. *Guidelines for the Development of Military Training Decision Aids*. NPRDC TR 88-16. Navy Personnel Research and Development Center, San Diego, California.

Mallary, T. C. 1985. "Design of the Human-Computer Interface for a Computer Aided Design Tool for the Normalization of Relations." Master's Thesis, Air Force Institute of Technology, Wright Air Force Base, Ohio.

Marcus, A., N. Smilonich, and L. Thompson. 1995. *The Cross-GUI Handbook*. Addison-Wesley Publishing Company, Reading, Massachusetts.

Marcus, A. 1992. *Graphic Design for Electronic Documents and User Interfaces*. ACM Press, New York.

Marcus, A. 1984. "Corporate Identity for Iconic Interface Design: The Graphic Design Perspective." *IEEE Computer Graphics and Applications (CG&A)* 4(12):24-32.

Marcus, A. 1979. *Managing Facts and Concepts*. National Endowment for the Arts, Washington, D.C.

Martin, J. 1983. *Managing the Data-Base Environment*. Prentice Hall, Englewood Cliffs, New Jersey.

Masiaszek, L. A. 1990. *Database Design and Implementation*. Prentice Hall, New York.

Matthews, M. L. 1987. "The Influence of Color on CRT Reading Performance and Subjective Comfort Under Operational Conditions." *Applied Ergonomics* 18.4:323-328.

McCann, C. 1988. Final Report from "Research Study Group 12 (RSG.12) on Computer-Human Interaction in Command and Control." Document AC/243 (Panel 8/RSG.12) D/7, DCIEM, Downsview, Ontario.

McCann, P. H. 1983. *Methods for Improving the User-Computer Interface*. Report No. NPRDC TR 83-29, Navy Personnel Research and Development Center, San Diego, California.

McKeown, P. E., J. J. Fallesen, M. S. Perkins, and C. G. Ross. 1991. *Operations Planning Tools (OPT) Functional Description*. U.S. Army Research Institute Product 91-09 (AD-A235 665), Fort Leavenworth, Kansas.

McNeese, M. D., and L. Katz. 1986. "Legibility Evaluation of a Large-Screen Display System Under Medium Ambient Illumination." In *Society for Information Display 86 Digest*, pp. 142-145. SID, New York.

Meyer, A. 1992. "Developing a Portable C++ GUI Class Library." *Dr. Dobbs Journal*, Nov 92, pp. 102-109.

Microsoft Corporation. 1992. *The Windows<sup>TM</sup> Interface: An Application Design Guide*. Microsoft Press, Redmond, Washington.

Microsoft Corporation. 1991a. *Microsoft<sup>®</sup> C/C++ ver 7.0 Tutorial*. Doc No. LN24772-1191, Microsoft Press, Redmond, Washington.

Microsoft Corporation. 1991b. *Microsoft<sup>®</sup> Windows<sup>TM</sup> Multimedia Authoring and Tools Guide*. Microsoft Press, Redmond, Washington.

Milgram, P., D. Drascic, and J. J. Grodski. 1991. "Enhancement of 3-D Video Displays by Means of Superimposed Stereo-Graphics." In *Proceedings of the Human Factors Society - 35th Annual Meeting*, pp. 1457-1461. HFS, Santa Monica, California.

Milheim, W.D., and B. L. Martin. 1991. "Theoretical Basis for the Use of Learner Control: Three Different Perspectives." *Journal of Computer-Based Instruction* 18(3):99-105.

Miller, P. J. 1991. *NMR Display Prototype Experiment Results*. Technical Report No. AD-A239 131, Defense Technical Information Center, Alexandria, Virginia.

Minasi, M. 1994. *Secrets of Effective GUI Design*. SYBEX, Inc., Alameda, California.

Minasi, M. 1990. "Bayes and Simple Expert Systems." *AI Expert*, pp. 13-15.

Mitchell, D. K., and K. P. Kysor. 1992. *A Preliminary Evaluation of the Prototype Tactical Computerized Interactive Display*. Technical Memorandum 2-92. U.S. Army Human Engineering Laboratory, Aberdeen Proving Ground, Maryland.

Mittal, S. 1985. "Knowledge Acquisition from Multiple Experts." *AI Magazine*, pp. 32-36.

Miyashita, T., and T. Uchida. 1990. "Cause of Fatigue and its Improvement in Stereoscopic Displays." *Proceedings of Society for Information Display*, 31(3):249-254. SID, New York.

Moran, T. P. 1981. "The Command Language Grammar: A Representation for the User of Interactive Computer Systems." *International Journal of Man-Machine Studies* 15:3-50.

Morse, R. S. 1985. "Glare Filter Preference: Influence of Subjective and Objective Indices of Glare, Sharpness, Brightness, Contrast and Color." In *Proceeding of the Human Factors Society - 29th Annual Meeting*, pp. 782-786. HFS, Santa Monica, California.

Muracka, T., M. Kawamura, and H. Uesako. 1989. "Readability on the Positive Type Liquid Crystal Display Devices with Multinumeral Influenced by the Irradiation Illuminances." In *Work With Computers: Organizational, Management, Stress and Health Aspects*, eds. M. J. Smith and G. Salvendy, pp. 542-548. Elsevier Science Publishers B.V., Amsterdam.

Murphy, T. 1993. "Looking at the World Through Cheap Sunglasses." *Computer Language* 10(2):63-85.

Muter, P., and C. Mayson. 1986. "The Role of Graphics in Item Selection from Menus." *Behaviour and Information Technology* 5(1):89-95.

National Institute of Standards and Technology (NIST). 1993. "User Interface Component of Applications Portability Profile." *Federal Information Processing Standard (FIPS) 158-1*. NIST, Gaithersburg, Maryland.

National Institute of Standards and Technology (NIST). 1991a. *Application Portability Profile (APP) The U.S. Government's Open Systems Environment Profile OSE/I, Version 1.0* (FIPS Pub 151-1), NIST Special Report 500-187, National Institute of Standards and Technology, Gaithersburg, Maryland.

National Institute of Standards and Technology (NIST). 1991b. *Government Open Systems Interconnection Profile (GOSIP), Version 1.0* (FIPS Pub 146-1), National Institute of Standards and Technology, Gaithersburg, Maryland.

National Institute of Standards and Technology (NIST). 1990. *POSIX, Portable Operating System Interface for Computer Environments (IEEE 1003.1-1988)* (FIPS Pub 151-1), National Institute of Standards and Technology, Gaithersburg, Maryland.

- Nes, F. 1986. "Space, Color, and Typography on Visual Display Terminals." *Behavior and Information Technology* 5(2):99-118.
- Neurath, O. 1980. *International Picture Language/Internationale Bildersprache*. University of Reading, England.
- Neuron Data, Inc. 1992a. *Open Interface Technical Overview*. Neuron Data, Inc., Palo Alto, California.
- Neuron Data, Inc. 1992b. *Open Interface Programmer's Guide, Version 2.0*. PN Man-30-400-02, Neuron Data, Inc, Palo Alto, California.
- Nickerson, R. S. 1986. *Using Computers: The Human Factors of Information Systems*. The MIT Press, Cambridge, Massachusetts.
- Nicol, A. 1990. "Interface for Learning: What Do Good Teachers Know That We Don't?" In *The Art of Human-Computer Interface Design*, ed. B. Laurel, pp. 113-122. Addison-Wesley, Reading, Massachusetts.
- Nielsen, J. 1987. "A User Interface Case Study of the Macintosh." In *Cognitive Engineering in the Design of Human-Computer Interaction and Expert Systems*. ed. G. Salvendy, pp. 241-248. Elsevier Science Publishers B.V., Amsterdam.
- Nielsen, J. 1990. *Hypertext and Hypermedia*. Academic Press, Inc., Boston.
- NIST - see National Institute of Standards and Technology
- Nolan, P. R. 1989. "Designing Screen Icons: Ranking and Matching Studies." In *Proceedings of the Human Factors Society 33rd Annual Meeting, Volume 1*, pp. 380-384. Human Factors Society, Santa Monica, California.
- Norman, K. L. 1991. *The Psychology of Menu Selection: Designing Cognitive Control of the Human/Computer Interface*. Ablex Publishing Corporation, Norwood, New Jersey.
- Norman, K. L., and J. P. Chin. 1988. "The Effect of Tree Structure on Search in a Hierarchical Menu Selection System." *Behavior and Information Technology* 7(1):51-65.
- North Atlantic Treaty Organization (NATO). 1990. *North Atlantic Treaty Organization Standardization Agreement 2019, Military Symbols for Land Based Systems*. U.S. Navy, Washington, D.C.
- O'Keefe, R. M. 1989. "The Evaluation of Decision-Aiding Systems: Guidelines and Methods." *Information and Management* 17:217-226.
- O'Malley, C., P. Smolensky, L. Bannon, E. Conway, J. Graham, J. Sokolov, and M. L. Montry. 1983. "A Proposal for User-Centered System Documentation." In HMI Project. *User-Centered System Design: Papers for the CHI '83 Conference on Human Factors in Computer Systems*, p.

4-8. DTIC No. AD 136131. (Technical Report). Office of Naval Research/Personnel and Training, Arlington, Virginia.

Ogden, W. C., and S. R. Brooks. 1983. "Query Languages for the Casual User: Exploring the Middle Ground between Formal and Natural Languages." In *Human Factors in Computing Systems: Proceedings of the CHI '83 Conference*, ed. A. Janda. North Holland Publishing Company, Amsterdam.

Olsen, L. A., and T. N. Huckin. 1991. *Technical Writing and Professional Communication (2nd ed.)*. McGraw-Hill, New York.

Olson, J. M. 1987. "Color and the Computer in Cartography." In *Color and the Computer*, ed. H. J. Durrett, pp. 205-219. Academic Press, Inc., San Diego, California.

Open Software Foundation (OSF). 1992. *OSF/Motif<sup>TM</sup> Style Guide*, Revision 1.2. Prentice Hall, Englewood Cliffs, New Jersey.

Osborn, P. B., and W. H. Zickefoose. 1990. "Building Expert Systems From the Ground Up." *AI Expert*, pp. 28-35.

OSF - see Open Software Foundation

Otte, F. H. 1982. "Consistent User Interface." In *Human Factors and Interactive Computer Systems*, pp. 261-275. Ablex, Norwood, New Jersey.

Owen, D. 1987. "Direct Manipulation and Procedural Reasoning." In *Cognitive Engineering in the Design of Human-Computer Interaction and Expert Systems*, ed. G. Salvendy, pp. 349-356. Elsevier Science Publishers, B.V., Amsterdam.

Paap, K. R., and R. J. Roske-Hofstrand. 1988. "Design of Menus." In *Handbook of Human-Computer Interaction*, ed. M. Helander, Elsevier Science Publishers, B.V., Amsterdam.

Paap, K. R., and R. J. Roske-Hofstrand. 1986. "The Optimal Number of Menu Options Per Panel." *Human Factors* 28(4):377-385.

Parker, S. P., ed. 1989. *McGraw-Hill Dictionary of Scientific and Technical Terms*. 4th ed. McGraw-Hill, New York.

Parkinson, S. R., M. D. Hill, N. Sisson, and C. Viera. 1988. "Effects of Breadth, Depth, and Number of Responses on Computer Menu Search Performance." *International Journal of Man-Machine Studies* 28:683-692.

Parrish, R. N., J. L. Gates, and S. J. Munger. 1981. *Design Guidelines and Criteria for User/Operator Transactions with Battlefield Automated Systems Volume IV: Provisional Guidelines and Criteria*. Technical Report 537, U.S. Army Research Institute for the Behavioral and Social Sciences, Alexandria, Virginia.



Parsaye, K., M. Chignell, K. Setrag, and H. Wong. 1990. "Intelligent Databases." *AI Expert*, Mar. 1990, pp. 38-47, Miller Freeman, Inc.

Patricia Seybold's Office Computing Group. 1989. "Patricia Seybold's Office Computing Report." April 1989. 12(4):1(9).

Pavard, B. 1987. "Design of Graphic Dialogue Without Syntactic Constraints." In *Cognitive Engineering in the Design of Human-Computer Interaction and Expert Systems*, ed. G. Salvendy, pp. 349-356. Elsevier Science Publishers B.V., Amsterdam.

Payne, S. J. 1983. "Readability of Liquid Crystal Displays: A Response Surface." *Human Factors Society, Inc.*, 25(2):185-190. Santa Monica, California.

Pejtersen, A. M., and L. P. Goodstein. 1990. "Beyond the Desk Top Metaphor: Information Retrieval With an Icon-Based Interface." *Visualization in Human-Computer Interaction, Interdisciplinary Workshop in Informatics and Psychology*. Springer-Verlag, Berlin.

Petrún, C. J., W. Hernon, and M. MacDonald. 1985. "An Examination of the Relative Legibility of Text on One Line Vacuum Florescent and Liquid Crystal Displays." In *Proceedings of the Human Factors Society - 29th Annual Meeting*, pp.1122-1124. HFS, Santa Monica, California.

Pew, R. W. 1988. "Human Factors Issues in Expert Systems." In *Handbook of Human-Computer Interaction*, ed. M. Helander, pp. 931-940. Elsevier Science Publishers B.V., Amsterdam.

Pleshko, P. 1991. "AC Electroluminescent Display Technology: Challenges and Potential." *Proceedings of Society for Information Display*, 32(2):106-108. SID, New York.

Rancourt, J., W. Grenawalt, M. W. Hunter, and H. L. Snyder. 1986. "Quantitative Evaluation of the Effect of an Antireflection Filter." In *Society for Information Display 86 Digest*, pp. 420-423. SID, New York.

Raum, H. G. 1988. *Design and Implementation of an Expert User Interface for the Computer Aided Prototyping System*. Thesis for U.S. Naval Postgraduate School, Monterey, California.

Raymond, E. S., ed. 1991. *The New Hacker's Dictionary*. The MIT Press, Cambridge, Massachusetts.

Reedy, A.E., D.L. Smith, and W.G. Bail. 1992. "Software Architecture Framework (Draft)," Defense Systems Information Agency/Center for Information Management, McLean, Virginia.

Reger, J. J., H. L. Snyder, W. W. Farley. 1989. "Legibility of Emissive and Non-Emissive Displays Under Florescent and Daylight Illumination." In *Society for Information Display 89 Digest*, pp. 364-367. SID, New York.



Reinhart, W. F. 1990. "Effects of Depth Cues on Depth Judgments Using a Field-Sequential Stereoscopic CRT Display." Unpublished Doctoral Dissertation, Virginia Polytechnic Institute and State University, Blacksburg, Virginia.

Relles, N. 1981. "A User Interface for On-line Assistance." In *Proceedings of the Fifth International Conference on Software Engineering*, pp. 400-408. Institute for Electrical and Electronics Engineers (IEEE), New York.

Relles, N., and L. A. Price. 1981. "A User Interface for On-line Assistance." In *Proceedings of the Fifth International Conference on Software Engineering*. Institute for Electrical and Electronics Engineers (IEEE), New York.

Rich, E. 1983. "Users Are Individuals: Individualizing User Models." *International Journal on Man-Machine Studies* 18:199-214.

Riedel, S. L. 1988. "User Acceptance and Field Implementation of Decision Support Systems." *Research Report 1477*. U.S. Army Research Institute Field Office, Fort Leavenworth, Kansas.

Ripley, G. D. 1989. "DVI--A Distal Multimedia Technology". *Communications of the ACM* 32:811-822.

Roach, J., R. Hartson, R. W. Ehrich, T. Yuntan, and D. H. Johnson. 1982. "DMS: A Comprehensive System for Managing Human-Computer Dialogue." In *Human Factors in Computer Systems: Proceedings* March 15-17, 1982, Gaithersburg, Maryland.

Rockley, A. 1987. "Online Documentation: From Proposal to Finished Product." In *Proceedings of 34th International Technical Communication Conference*, ATA 58-61. Society for Technical Communication, Washington, D.C.

Rogers, Y. 1989. "Icons at the Interface: Their Usefulness." *Interacting With Computers - The Interdisciplinary Journal of Human-Computer Interaction* 1(1):105-117.

Rosch, W. L. 1994. *The Winn L. Rosch Hardware Bible*, 3rd Edition. Brady Publishing, Indianapolis, Indiana.

Rosenborg, V. 1993. *A Guide To Multimedia*. New Riders Publishing, Carmel, Indiana.

Rosenstein, M., and L. Weitzman. 1990. "The HITS Icon Editor - the Specification of Graphic Behavior Without Coding." *IEEE*, 0073-1129/90/0523:523-529.

Roth, J.L., J. A. Fitzpatrick, R. E. Warm, and J. L. Ditzian. 1988. Implementing *Embedded Training (ET): Volume 5 of 10: Designing the ET Component*. ARI Research Product No. 88-28, NTIS No. AD-A205 697. U.S. Army Research Institute for the Behavioral and Social Sciences, Alexandria, Virginia.

Rupp, B. A. 1981. "Visual Display Standards: A Review of Issues." *Proceedings of Society for Information Display*, 22(1):63-72. SID, New York.

Salvendy, G. 1987. *Cognitive Engineering in the Design of Human-Computer Interaction and Expert Systems*. Elsevier Science Publishers B.V., Amsterdam.

Schauer, U. 1983. "The Integrated Data Analysis and Management System - A Generator for Enduser Systems." *Enduser Systems and Their Human Factors* eds. A. Blaser and M. Zoeppriz, pp. 30-61. Springer-Verlag, Berlin.

Schmitz, J. D., G. D. Armstrong, and J. D. Little. 1990. "Cover Story - Automated News Finding in Marketing." *Interfaces* 20(6):29-38.

Schur, S. 1988. "The Intelligent Database." *AI Expert*, Jan. 1988, pp. 26-34, Miller Freeman, Inc.

Schwartz, J. P. 1983. "Lack of Guidance for Decision Aid Interface Design." *SIGCHI Bulletin* 15:13-17.

Sellen, A., and A. Nicol. 1990. "Building User-Centered On-line Help." In *The Art of Human-Computer Interface Design*, ed. B. Laurel, pp. 143-153. Addison-Wesley, Reading, Massachusetts.

Seyer, P. 1991. *Understanding Hypertext Concepts and Applications*. Windcrest Books, Blue Ridge Summit, Pennsylvania.

Shaw, B., and M. McCauley. 1985. *Personal Computer Dialogue: A Human Engineering Data Base Supplement*. Technical Report No. AFAMRL-TR-85-013. U.S.A.F. Aerospace Medical Research Laboratory, San Antonio, Texas.

Shields, S. E., and W. P. Bleha. 1991. "Liquid Crystal Light Valves for Projection Displays." In *Society of Photoptic Instrumentation Engineers*, 1455:225-236. SPIE Press, Bellingham, Washington.

Shneiderman, B. 1992. *Designing the User Interface: Strategies for Effective Human-Computer Interaction*, Second Edition. Addison-Wesley Publishing Company, Reading, Massachusetts.

Shneiderman, B. 1991. "Visual User Interfaces for Information Exploration," made available at the TAE Ninth User's Conference, New Carrollton, Maryland. CAR-TR-577, CS-TR-2748, Human-Computer Interaction Laboratory and Department of Computer Sciences, University of Maryland, College Park, Maryland.

Shneiderman, B. 1988. "We Can Design Better User Interfaces: A Review of Human-Computer Interaction Styles." *Ergonomics* 31(5):699-710.

Shneiderman, B. 1987. *Designing the User Interface: Strategies for Effective Human-Computer Interaction*. Addison-Wesley, Reading, Massachusetts.

Shneiderman, B. 1984. "The Future of Interactive Systems and the Emergence of Direct Manipulation." In *Human Factors and Interactive Computer Systems*, ed. Y. Vassiliou, pp. 1-27. Ablex Publishing Corporation, Norwood, New Jersey.

Shneiderman, B. 1982a. "Designing Computer System Messages." *Communications of the ACM*, 25(9):604-605.

Shneiderman, B. 1982b. "Human Factors Experiments in Designing Interactive Systems." *Tutorial: End User Facilities in the 1980's*, ed. J. A. Larson, pp. 16-26. Proceedings of the IEEE Computer Society Sixth International Computer Software & Applications Conference, IEEE Computer Society Press, New York.

Shneiderman, B. 1978. "Improving the Human Factors Aspect of Database Interactions." *ACM Transactions on Database Systems* 3(4):417-439.

Shneiderman, B., and G. Kearsley. 1989. *Hypertext Hands-On! An Introduction to a New Way of Organizing and Accessing Information*. Addison-Wesley Publishing Company, Reading, Massachusetts.

Shurtleff, D. A., W. F. Wuersch, and J. G. Rogers. 1982. "Applications of Large-Screen Display Legibility Criteria." In *Society for Information Display 82 Digest*, pp. 202-203. SID, New York.

Shurtleff, D. A., W. F. Wuersch, and J. G. Rogers. 1981. "How to Make Large Screen Displays Legible." In *Proceedings of the Human Factors Society - 25th Annual Meeting*, pp. 149-153. HFS, Santa Monica, California.

Sidorsky, R. 1984. *Design Guidelines for User Transactions with Battlefield Automated Systems: Prototype for a Handbook*. DTIC AD-A153 231, U.S. Army Research Institute for the Behavioral and Social Sciences, Alexandria, Virginia.

Silverstein, L. D., R. W. Monty, J. W. Huff, and K. L. Frost. 1987. *Image Quality and Visual Simulation of Color Matrix Displays*. Technical Paper No. 871789, The Engineering Society for Advancing Mobility Land, Sea, Air, and Space, Long Beach, California.

Sisson, N., S. R. Parkinson, and K. Snowberry. 1986. "Consideration of Menu Structure and Communication Rate for the Design of Computer Menu Displays." *International Journal of Man-Machine Studies* 25:479-489.

Slominski, S.E., and I. R. Young. 1988. *A User Friendly Design of an Interactive Prototype for the Maintenance and Monitoring of Civilian Training Records*. Unpublished Master's Thesis, Naval Postgraduate School, Monterey, California.

Smith, M. C., and L. E. Magee. 1980. "Tracing the Time Course of Picture-Word Processing." *Journal of Experimental Psychology: General* 109(4):373-392.

Smith, S. L., and J. N. Mosier. 1986. *Guidelines for Designing User Interface Software*. The MITRE Corp, Bedford, Massachusetts.

Smith, S. L., and J. N. Mosier. 1984. *A Design Evaluation Checklist for User Computer Interface Software*. The MITRE Corp., Bedford, Massachusetts.

Snyder, H. L. 1988. "Image Quality." In *Handbook of Human-Computer Interaction*, ed. M. Helander, pp. 437-474. Elsevier Science Publishers B.V., Amsterdam.

Snyder, H. L. 1987a. "The ANSI Human Factors Standard for Visual Display Terminal Workstation: A Process and Progress Report (Invited Address)." In *Society Information Display 87 Digest*, pp. 14-17. SID, New York.

Snyder, H. L. 1987b. "Counterintuitive Criteria for Visual Displays." In *Social, Ergonomic and Stress Aspects of Work with Computers*, eds. G. Salvendy, S. L. Sauter, and J. J. Hurrell, Jr., pp. 145-156. Elsevier Science Publishers B. V., Amsterdam.

Snyder, H. L. 1984. "Ergonomics Database for Visual Displays and VDUs." In *Ergonomic Data for Equipment Design*, ed. H. Schmidtke, pp. 219-234. Plenum Press, New York.

Snyder, H. L. 1980. "Human Visual Performance and Flat Panel Display Image Quality." HFL-80-1/ONR-80-1, Human Factors Laboratory, Virginia Polytechnic Institute and State University, Blacksburg, Virginia.

Sormunen, E. 1987. "A Knowledge-Based Intermediary System for Information Retrieval," in *Knowledge Engineering Expert Systems and Information Retrieval*, ed. I. Wormell, pp. 59-73. Taylor Graham, London.

Spence, R., and M. Parr. 1991. "Cognitive Assessment of Alternatives." *Interacting With Computers* 3(3):270-282.

Stammers, R. B., and J. Hoffman. 1991. "Transfer Between Icon Sets and Ratings of Icon Concreteness and Appropriateness." In *Proceedings of the Human Factors Society 35th Annual Meeting*, Vol. 1, pp. 354-358. San Francisco.

Steele, C. A., J. Harrold, J. P. Stanton, and R. Daley. 1987. "Performance of Laser-Addressed Liquid Crystal Map Overlay Display." In *Society of Photoptic Instrumentation Engineers*, 760:70-73. SPIE Press, Bellingham, Washington.

Strategic Air Command (SAC). 1990. *Intelligence Data Handling System (IDHS) Man-Machine Interface (MMI) Style Guide, Version 1.0*. Washington, D.C.

Sun Microsystems, Inc. 1990. *The Open Look Graphical User Interface Application Style Guidelines*. AT&T, New Jersey.

Suntola, T. 1989. "VLSI and Computer Peripherals: Thin Film EL-Display," *Proceedings of the Institution of Electrical and Electronics Engineers*, pp. 2-32/2-35. IEEE Computer Society Press, Washington, D.C.

Tannas, L. E., ed. 1985. *Flat-Panel Displays and CRTs*. Van Nostrand Reinhold, New York.

Tannenbaum, A. 1990. "Installing AI Tools Into Corporate Environments." *AI Expert*, pp. 54-59.

Tenopir, C., and G. Lundeen. 1988. *Managing Your Information*. Neal-Schuman Publishers, New York.

Thierauf, R. J. 1988. *User-Oriented Decision Support Systems*. Prentice Hall, Englewood Cliffs, New Jersey.

Thomas, J. C. 1983. "Psychological Issues in Data Base Management." In *Designing for Human-Computer Communications*, eds. M. E. Sime and M. J. Coombs, pp. 169-184. Academic Press, London.

Thomas, J.C., and J. M. Carroll. 1981. "Human Factors in Communication." *IBM Systems Journal* 20(2):236-263.

Thorell, L. G., and W. J. Smith. 1990. *Using Computer Color Effectively: An Illustrated Reference*. Prentice Hall, Englewood Cliffs, New Jersey.

TRW. 1990a. *The AWIS Common User/System Interface Style Guide*. AWIS Program Documentation.

TRW. 1990b. *UTACCS, Soldier-Machine Development Standards*. UTACCS Program Documentation.

Tullis, T. S. 1988. "Screen Design." In *Handbook of Human-Computer Interaction*, ed. M. Helander, pp. 377-411. Elsevier Science Publishers, B.V., Amsterdam.

Tullis, T. S. 1981. "An Evaluation of Alphanumeric, Graphic, and Color Information Displays." *Human Factors* 23(5):541-550.

Tzeng, O.C.S., N. T. Trung, and R. W. Rieber. 1990. "Cross-Cultural Comparisons on Psychosemantics of Icons and Graphics." *International Journal of Psychology* 25:77-97.

U.S. Department of the Army. 1987. *Map Reading and Land Navigation*. FM 21-26, Washington, DC.

U.S. Department of the Army. 1985a. *Authorized Abbreviations and Brevity Codes*. AR310-50, Army UPDATE Publications, Washington, D.C.

U.S. Department of the Army. 1985b. *Operational Terms and Symbols*. FM 101-5-1, U.S. Army Combined Arms Center, Fort Leavenworth, Kansas.

U.S. Department of the Army. 1984. *Human Engineering Guide to Equipment Design*. FM 21-26 (sec 11), eds. H. P. Van Cott and R. G. Kinkade. John Wiley & Sons, New York.

U.S. Department of Defense (DoD). 1994. *Common Warfighting Symbolology, Version 1*. MIL-STD-2525, U.S. Department of Defense, Washington, D.C.

U.S. Department of Defense (DoD). 1992a. *Department of Defense Human Computer Interface Style Guide* (Version 1.0), Defense Information Systems Agency/Center for Information Management, McLean, Virginia.

U.S. Department of Defense (DoD). 1992b. *Department of Defense Human Computer Interface Style Guide* (Version 2.0), Defense Information Systems Agency/Center for Information Management, McLean, Virginia.

U.S. Department of Defense (DoD). 1992c. "Standards-Based Architecture Planning Handbook, Draft Version 1.0." Produced by DMR Group, Inc. for DISA XI, Washington, D.C.

U.S. Department of Defense (DoD). 1992d. *Technical Reference Model for Information Management, Version 1.2*, Defense Information Systems Agency/Center for Information Management, McLean, Virginia.

U.S. Department of Defense (DoD). 1991a. *Department of Defense Intelligence Information Systems Style Guide*, Department of Defense Intelligence Information Systems (DODIIS) Management Board, Arlington, Virginia.

U.S. Department of Defense (DoD). 1991b. *DoD Directive 5000.1 Defense Acquisition*, U.S. Department of Defense, Washington, D.C.

U.S. Department of Defense (DoD). 1991c. *DoD Instruction 5000.2. Defense Acquisition Management Policy and Procedures*, U.S. Department of Defense, Washington, D.C.

U.S. Department of Defense (DoD). 1991d. *Air Crew Station Alerting Systems*. MIL-STD-411E, U.S. Department of Defense, Washington, D.C.

U.S. Department of Defense (DoD). 1990. *Military Training Programs*. MIL-STD-1379D, Naval Sea Systems Command, SEA 55Z3.

U.S. Department of Defense (DoD). 1989a. *Department of Defense Dictionary of Military and Associated Terms*. DoD, Washington, D.C.

U.S. Department of Defense (DoD). 1989b. *Human Engineering Design Criteria for Military Systems, Equipment, and Facilities*. MIL-STD-1472D, U.S. Army Missile Command, Huntsville, Alabama.

U.S. Department of Defense (DoD). 1989c. *Human Engineering Guidelines for Management Information Systems*. DOD-HDBK-761A, DoD, Washington, D.C.

U.S. Department of Defense (DoD). 1988. *DoD Directive 7920.1. Life-Cycle Management of Automated Information Systems (AISs)*, U.S. Department of Defense, Washington, D.C.

U.S. Department of Defense (DoD). 1984. *Legends for Use in Air Crew Stations and on Airborne Equipment*. MIL-STD-783D, U.S. Department of Defense, Washington, D.C.

U.S. Department of Defense (DoD). 1981. *Abbreviations for Use on Drawings, Specification Standards, & in Technical Documents*. MIL-STD-12D, U.S. Department of Defense, Washington, D.C.

Ullman, J. D. 1988. *Principles of Database and Knowledge-Base Systems*. Computer Science Press, Rockville, Maryland.

UNIX System Laboratories, Inc. 1991. *Open Look Graphical User Interface User's Guide*. Prentice Hall, Inc., Englewood Cliffs, New Jersey.

Urban, C. D. 1990. "Design and Evaluation of a Tactical Decision Aid." *Proceedings IEEE International Conference on Systems, Man, and Cybernetics*. DTIC No. AD-A232 001, IEEE, New York.

Valdes, R. 1992a. "Sizing Up Application Frameworks and Class Libraries." *Dr. Dobbs's Journal* (Oct 92) pp. 18-27.

Valdes, R. 1992b. "Sizing Up GUI Toolkits." *Dr. Dobbs Journal* (Nov 92), pp. 18-26.

Van Cott, H. P., and R. G. Kinkade, eds. 1984. *Human Engineering Guide to Equipment Design*. John Wiley & Sons, New York.

Vassiliou, Y., ed. 1984. *Human Factors and Interactive Computer Systems*. Ablex Publishing Corporation, Norwood, New Jersey.

Vance, D. W. 1987. "Image Distortions in Video Projection." In *Large Screen Projection Displays*, 760:54-59. SPIE Press, Bellingham, Washington.

Vasta, J. A. 1985. *Understanding Data Base Management Systems*. Wadsworth Publishing Company, Belmont, California.

Vaughan, T. 1993. *Multimedia: Making It Work*. Osborne McGraw-Hill, Berkeley, California.

Veith, R. H. 1988. *Visual Information Systems: The Power of Graphics and Video*. G.K. Hall & Co., Boston.



- Veron, H., D. A. Southard, J. R. Leger, and J. L. Conway. 1990. *3D Displays for Battle Management*. Technical Report No. AD-A223 142, Defense Technical Information Center, Alexandria, Virginia.
- Veron, H. 1988. *A Resolution Measurement Technique for Large Screen Displays*. Technical Report No. M88-59, The MITRE Corp., Bedford, Massachusetts.
- Verplank, W. L. 1988. "Graphic Challenges in Designing Object-Oriented User Interfaces." In *Handbook of Human-Computer Interaction*, ed. M. Helander, pp. 365-375. Elsevier Science Publishers B.V., Amsterdam.
- Visix Software, Inc. 1993. *Galaxy Resource Builder User's Guide, Version 1.2*. Visix, Software, Inc., Reston, Virginia.
- Visix Software, Inc. 1992. *Galaxy Application Environment Technical Description*. Visix Software, Inc., Reston, Virginia.
- Waern, Y., and C. Rollenhagen. 1983. "Reading Text From Visual Display Units (VDUs)." In *International Journal of Man-Machine Studies* 18, Academic Press, London.
- Walker, J. H. 1987. "Issues and Strategies for Online Documentation." *IEEE Transactions on Professional Communication*, PC-30 (4):235-248.
- Walrath, J. D. 1989. "Aiding the Decision-Maker: Perceptual and Cognitive Issues at the Human-Machine Interface." *Technical Note 15-89*. U.S. Army Human Engineering Laboratory (DTIC No. AD-A217 862), Aberdeen Proving Ground, Maryland.
- Wehrer, W., and E. E. Mitchamore. 1992. "Technology/Marketing Case Study: Plug-and-play IR Touch Screens." *Information Display*, pp. 10-12. SID, New York.
- Weinschenk, S., and S. C. Yeo. 1995. *Guidelines for Enterprise-Wide GUI Design*. John Wiley & Sons, New York.
- Weingaertner, S. T., and A. H. Levis. 1988. "Evaluation of Decision Aiding in Submarine Emergency Decision-Making." IFAC Conference on the Analysis, Design and Evaluation of Man-Machine Systems, Oulu, Finland, pp. 195-201.
- Wenger, E. 1987. *Artificial Intelligence and Tutoring Systems*. Morgan Kaufman Publishers, Los Altos, California.
- Wexelblat, R. L. 1989. "On Interface Requirements for Expert Systems." *AI Magazine* 10:66-78.
- Wiebe, A. F., D. Johnson, G. A. Harcey, and M. Teter. 1993. "The Task-Training Guideline: A Powerful Format for How-To Instructional Training Materials." *STC Technical Communication* 40(1):49-61.



Williams, K.E., C. J. Hamel, and L. B. Shrestha. 1987a. *An Evaluation of Characteristics Contributing Towards Ease of User-Computer Interface in a Computer-Aided Instruction Exercise*. Technical Report TR87-030, Naval Training Systems Center, Orlando, Florida.

Williams, K.E., C. J. Hamel, and L. B. Shrestha. 1987b. *CAI Evaluation Handbook: Guidelines for the User Interface Design for Computer-Aided Instruction*. Technical Report No. TR87-033, Naval Training Systems Center, Orlando, Florida.

Williams, R. D. 1990. "Volume 3D Display Technology." In *Stereographics*, 17th International Conference on Computer Graphics and Interactive Techniques, Association for Computing Machinery, Special Interest Group on Graphics, Dallas, Texas.

Wilson, M., P. Barnard, and A. MacLean. 1990. An Investigation of the Learning of a Computer System. In *Cognitive Ergonomics: Understanding, Learning and Designing Human-Computer Interaction*, ed. P. Falzon, pp. 151-172. Academic Press, London.

Witten, I. H., and S. Greenberg. 1985. "Adaptive Personalized Interfaces -- A Question of Viability." *Behavior and Information Technology* 4(1):31-45.

Wodaski, R. 1992. *Multimedia Madness!* Sams Publishing, Carmel, Indiana.

Wood, W. T., and S. K. Wood. 1987. "Icons in Everyday Life." In *Advances in Human Factors/Ergonomics, 10 A -- Proceedings of the Second International Conference on Human-Computer Interaction - Social, Ergonomic and Stress Aspects of Work with Computers*, eds. G. Salvendy, S. L. Sauter, and J. J. Hurrell, Jr., pp. 97-105, Honolulu, Hawaii.

Woodson, W. E., B. Tillman, and P. Tillman. 1992. *Human Factors Design Handbook: Information and Guidelines for the Design of Systems, Facilities, Equipment, and Products for Human Use*, Second Edition. McGraw-Hill, Inc., New York.

WordPerfect Corporation. 1991. *WordPerfect for Windows<sup>TM</sup>, Reference*. WordPerfect Corporation, Orem, Utah.

Wormell, I., ed. 1987. *Knowledge Engineering Expert Systems and Information Retrieval*. Taylor Graham, London.

Wyatt, A. L. 1990. *Computer Professional's Dictionary*. Osborne McGraw-Hill, Berkeley, California.

XVT Software, Inc. 1992. *XVT Portability Toolkit Manual*. XVT Software, Inc., Boulder, Colorado.

Yamazaki, T., K. Kamijo, and S. Fukuzumi. 1990. "Quantitative Evaluation of Visual Fatigue Encountered in Viewing Stereoscopic 3D Displays: Near-Point Distance and Visual Evoked Potential Study." *Proceedings of Society for Information Displays*, 31(3):245-247. SID, New York.

Yeh, Y., and L. D. Silverstein. 1991. "Human Factors for Stereoscopic Color Displays." In *Society for Information Display 91 Digest*, pp. 826-829. SID, New York.

Yeh, Y., and L. D. Silverstein. 1989. "Using Electronics Stereoscopic Color Displays: Limits of Vision and Depth Discrimination." *Three-Dimensional Visualization and Display Technologies*, Proceedings of SPIE International Society for Optical Engineering, Vol. 1083, pp. 196-204. SPIE Press, Bellingham, Washington.

Zachary, W. W. 1988. "Decision Support Systems: Designing to Extend the Cognitive Limits." In *Handbook of Human-Computer Interaction*, ed. M. Helander, pp. 997-1030. Elsevier Science Publishers B.V., Amsterdam.

Ziegler, J. E., and K. P. Fährnich. 1988. "Direct Manipulation." In *Handbook of Human-Computer Interaction*, ed. M. Helander, pp. 123-134. Elsevier Science Publishers B.V., Amsterdam.

Zmud, R. W. 1978. "Concepts, Theories and Techniques: An Empirical Investigation of the Dimension of the Concept of Information." *Decision Sciences* 9:187-195.